Pruning 3D Filters For Accelerating 3D ConvNets

Zhenzhen Wang, Weixiang Hong, Yap-Peng Tan, *Fellow, IEEE*, and Junsong Yuan, *Senior Member, IEEE*

***Abstract*—Many methods have been proposed to accelerate 2D ConvNets by removing redundant parameters. However, few efforts are devoted to the problem of accelerating 3D Convolutional Networks. The 3D ConvNets, which are mainly designed for extracting spatiotemporal features, have been widely used in many video analytics tasks, such as action recognition and scene analysis. In this paper, we focus on accelerating 3D ConvNets for two motivations: (1) Fast video processing techniques are in dire need due to the explosive growth of video data; (2) Compared with individual images, video data consist of consecutively similar frames, thus are inherently more redundant. In this paper, we present a novel algorithm to dramatically accelerate 3D ConvNets by pruning redundant convolutional filters, while preserving the discriminative power of the networks. Specifically, we formulate the filter pruning from 3D ConvNets as a subset selection problem where each filter is regarded as a candidate. Determinantal Point Processes (DPPs) are employed to discriminatively select the filter candidates which are informative and yet diverse. We evaluate our method using two popular 3D networks, C3D and Pseudo-3D, on Sports-1 M dataset for video classification. Extensive experimental results demonstrate both the efficiency and performance advantages of our method. We also show that the proposed method can be easily generalized to 2D ConvNets pruning with promising experimental results on VGGnet and ResNet.**

***Index Terms*—3D ConvNets, Filter Pruning, DPPs, Maximum Abs. of Filters (MAF).**

I. Introduction

Deep convolutional networks (CNNs) have evolved to become the state-of-the-art techniques for many computer vision and multimedia tasks. The performance is generally enhanced with deeper and wider network architecture. However, as networks expand, their memory and computational cost also increase rapidly, making them difficult to be deployed on scenarios with computing or power constraint, such as mobile devices. According to [1], the memory cost is dominated by fully connected layers, while the computational cost mainly results from convolutional layers. For example, the fully connected layers of VGG-16 account for 90% of the total parameters while the convolutional layers contribute to more than 99% of the overall floating point operations (FLOPs). With the prevalence of mobile devices, the storage and computation requirements are both critical problems. The difficulties caused by huge storage cost have been well addressed by many efficient works, e.g., [2], [3]. Unfortunately, works on accelerating ConvNets by reducing FLOPs are limited and they mainly focus on 2D ConvNets [1], [4], [5], making 3D ConvNets pruning a more critical problem.

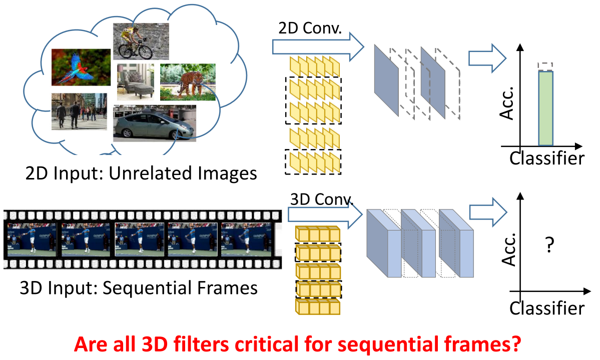

2D Conv.
2D Input: Unrelated Images
Acc.
Classifier
3D Conv.
3D Input: Sequential Frames
Acc.
?
Classifier
Are all 3D filters critical for sequential frames?

Fig. 1. Filter pruning in 2D ConvNets has shown excellent performance due to the redundant parameters in ConvNets, even though the inputs for 2D ConvNets are unrelated images. Will 3D ConvNets, whose inputs are sequential frames, be even more redundant?

In this work, we tackle the problem of accelerating 3D ConvNets, which is widely applied to action recognition [6], [7] and video understanding [8]–[10]. Compared with 2D ConvNets, accelerating 3D ConvNets is more critical for 2 reasons: 1) video data are inherently more redundant than individual images, thus it provides great potential to select effective filters from a large group of candidates; 2) reducing video data processing time is even more critical due to the huge video data size. The main difference from 2D ConvNets is that convolution filters of 3D ConvNets are 3-dimensional, i.e., (k, k, d) with the third dimension d denoting the filter temporal depth, so that the temporal information could be captured. Fig. 1 shows the difference between 2D ConvNets and 3D ConvNets.

The success of existing 2D ConvNets pruning methods illustrates remarkable redundancy in weights. For example, after being pruned off 64% parameters, VGG-16 exhibits only 0.15% extra errors on Cifar10 compared to the vanilla model [1]. In [5], it reports 2× speed-up on ResNet and Xception with only 1.4% and 1.0% accuracy drops, respectively. Since the consecutive frames in videos usually contain duplicate information, we hypothesize that the redundancy in the weights of 3D ConvNets is

Manuscript received February 15, 2019; revised July 4, 2019 and October 13, 2019; accepted October 25, 2019. Date of publication October 30, 2019; date of current version July 24, 2020. This work is supported in part by the start-up funds at University at Buffalo. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ramazan S. Aygun. *(Corresponding author: Zhenzhen Wang.)*

Z. Wang and Y.-P. Tan are with the School of Electrical and Electronic Engineering, Nanyang Technological University (NTU), Singapore 639798 (e-mail: zwang033@e.ntu.edu.sg; eyptan@ntu.edu.sg).

W. Hong is with the Ant Financial Service Group, Hangzhou 310000, China (e-mail: hwx229374@antfin.com).

J. Yuan is with the Department of Computer Science and Engineering, University at Buffalo, State University of New York, Clayton, NY 13624 USA (e-mail: jsyuan@buffalo.edu).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2019.2950523

more significant than that of 2D ConvNets. Moreover, accelerating 3D ConvNets is in dire need in practice due to the explosive growth of video data. Thus, we aim to accelerate 3D networks by 3D ConvNets pruning without significant performance drop.

Network pruning in CNNs can be conducted in a number of ways, such as layer-wise removal, feature map reduction and filter pruning. The layer-wise removal tailors the depth of the network and converts a deep network into a shallow one. Although this method is straightforward in implementation, it usually suffers from significant performance drop. Feature map reduction and filter pruning are at the same level of granularity. The former is to remove the redundant feature maps, so that the corresponding convolution operations are removed. The latter prunes the entire filters of $k \times k \times n_\ell$ in 2D ConvNets or $k \times k \times d \times n_\ell$ in 3D ConvNets, so that the feature maps in the following layers are also pruned. For a pretrained network, the filters are fixed but the feature maps are data-dependent. If a network is compressed by pruning the feature maps generated from input samples, then the pruned model needs to be updated once unseen samples are available. In such a case, the computational cost is still a burden. Filter pruning is therefore a better choice since it is neither too coarse in granularity nor too costly in computation. Moreover, filter pruning is a structured way of pruning without introducing the sparsity. Therefore it does not require sparse libraries or any specialized hardware.

To prune the redundant filters, we consider the redundancy in the following two cases: (1) filters are relatively uninformative, and (2) the functionality of a filter can be mimicked by other filters. Using the first criterion, the filters which are more informative than the rest are selected so that the accuracy drop resulted from filter pruning could be minimized. The second criterion is utilized to suppress duplicate filters so that the number of selected filters could be small enough to provide appreciable acceleration. By integrating these two criteria for filter selection, the pruned network can be accelerated dramatically while preserving the discriminative power. To fulfill this goal, we formulate the filter pruning problem as a subset selection problem where each filter is regarded as a candidate with a confidence score indicating its informativeness. Our purpose is to select the candidates that are both informative and mutually exclusive, and the diversity among filters can be enforced by selecting filters that are dissimilar to each other. To measure the similarity, we are inspired by [11] to describe each filter by a feature vector. Specifically, an n_ℓ-dimensional vector is generated by extracting the max absolute value of a filter from each $k \times k$ filter slice in 2D ConvNets or $k \times k \times d$ filter cube in 3D ConvNets. The resulting feature vector is termed as Maximum Abs of Filters (MAF). And the mutual similarities among filters can be obtained from MAFs through existing measurement, such as cosine similarity.

Specifically, the objective function is optimized by applying the Determinantal Point Processes (DPPs) to filters, which has been extensively used in diverse sampling problem [12]. To empirically evaluate the proposed method, we apply it to two popular 3D networks, C3D [13] and P3D [14], for the task of video classification on dataset Sports-1M [15]. For a fair comparison with existing methods, we also extend our proposed method to 2D filter pruning and show the results on Cifar10 [16], ImageNet [17] using VGG-16 [18] and ResNet [19] in terms of various pruning ratios. Our contributions are summarized as follows:

- Our proposed filter pruning method focuses on 3D ConvNets compression. Particularly, we formulate it as a subset selection problem by regarding each filter as a candidate.
- We introduce DPPs as a flexible and powerful tool for modeling filter level importance and diversity. A practical method is also presented for creating DPPs over filters.
- A filter representation, Maximum Abs. of Filters (MAF), is proposed for efficient similarity measurement, which is compact and translation invariant.
- We validate the effectiveness of our proposed method on accelerating two popular 3D networks for video classification and it is also generalized and applied to 2D network.

The remaining of the paper is organized as follows: Section II briefly reviews the pruning methods in the literature and also the sampling algorithm, DPPs [12]. Section III describes the proposed approach, i.e., the problem formulation and the optimization process, in detail. Experimental results and implementation details on two 3D networks for video classification, and the extension 2D pruning are presented in Section IV. Finally, we conclude this paper in Section V as an excellent method for subset selection.

II. Related Work

In this section, we first summarize network pruning techniques from two research directions: structured connection pruning and irregular pruning. The first is to remove the entire filters or feature maps to compress the networks, while the latter is to remove any unimportant connections/weights between layers to make the pruned networks sparse. The structured pruning methods can further be sub-categorized as weights-based pruning and activation-based pruning. Subsequently, we introduce the Determinantal Point Processes (DPPs) as optimization method for our proposed pruning goal.

Structured Pruning: Several works have studied removing feature maps or entire filters from a well-trained network. The most prevailing method is weights/filter based pruning. In [20], a cluster-center like method is proposed to select the filters. The concurrent work [21] measures the importance of a filter by the accuracy drop when removing it from the network. [22] proposes the depth multiplier method to scale down the number of filters in each convolutional layer by a factor. Lebedev and Lempitsky [23] use group-sparsity on the convolutional filters to achieve structured brain damage. Wen *et al.* [24] add structured sparsity regularizer on each layer to reduce trivial filters. [25] and [26] prune the redundant filters by adding an regularization term on the whole network to the loss function. [27] learns slim networks by identifying insignificant channels and removing them during training. [28] enforces structured sparsity to the network via introducing additional regularizers. [29] achieves network compression via layer quantization on different levels. In [30], Ye *et al.* force the outputs of some channels to be constant, and then to prune them by adjusting the biases of their impacting layers.

There are also many activation/feature maps based pruning methods. Polyak and Wolf [31] detect the less frequently activated feature maps with sample input data for face detection applications. Xavier *et al.* [32] apply PCA on feature maps to select the principal ones. Li *et al.* [33] quantitate the feature map importance in a feature-agnostic manner to guide model compression. In [34], a subset of feature maps that has the minimum reconstruction error with initial ones is selected. The advantage of structured pruning methods is that they are hardware independent and can be easily applied to standard DCNNs building blocks. But a further retraining process is usually required for reducing the performance loss.

Irregular Connection Pruning: A common approach to reduce the number of parameters is to remove the connections between layers. [35] and [36] propose to remove connections using information drawn from the Hessian of the network's error function. Sainath *et al.* [37] reduce the number of parameters by analyzing the weight matrices and applying low-rank factorization to the final weight layer. Han *et al.* [38] remove connections with weights smaller than a given threshold. DIVNET [39] seeks diversity among connections and merges similar neurons. Recently, Han *et al.* [40] alternate by pruning near-zero weights, which are encouraged by ℓ_1 or ℓ_2 regularization, and retraining the pruned networks. This line of work preforms connection pruning that leads to irregular network architectures. Thus, these techniques usually require complex sparse representation and delicate hardware customization for practical computational benefits.

Determinantal Point Processes (DPPs): A determinantal point process is a random point process which is for modelling repulsion. In machine learning, the focus of DPP-based models has been on diverse subset selection from a discrete and finite base set. It was demonstrated in [12] that DPP sampling, which is to select a subset with maximum determinant (det), has superior performance compared with existing subset selection approaches. More importantly, DPP sampling naturally fulfils our expectations for filter pruning, i.e., the selected filters should be informative and also diverse to each other. Particularly, each point in the distribution is assigned with a magnitude to represent its importance and a direction to measure the similarity to other points. Li *et al.* [41] further propose a fixed size subset selection, k-DPP. As its name implies, k-DPP will select a subset with fixed number of points, e.g., k, with which we are able to set the filter pruning ratios manually so that a trade-off between network compression ratio and performance loss can be achieved.

III. Our Method

Notations: Consider a well-trained 3D ConvNets with N layers, the weights of ℓ-th layer are parameterized by a 5-dimensional tensor $W^\ell \in \mathbb{R}^{n_{\ell+1}\times n_\ell \times k\times k\times d}$, where $n_{\ell+1}$ is the number of output channels; n_ℓ represents the number of input channels, and $k\times k\times d$ is the size of a 3-dimensional filter cube. For better illustration, we arrange the 5-dimensional tensor as a matrix, denoted as $\mathcal{F}_\ell$, with each element corresponding to a filter cube and each column corresponding to an entire filter, $\mathcal{F}_{\ell,i}$ (see Fig. 2). Then our purpose can be easily interpreted, i.e., to delete the redundant filters in ℓ-th convolutional layers (gray column in Fig. 2) and the corresponding feature maps (in gray) in $(\ell+1)$-th layer. As a result, the pruned network is expected to be computationally less expensive with minimum accuracy drop.

A. Filter Pruning via DPPs

Our goal is to select a subset of filters that are informative yet discriminative to each other, which is consistent with the goal of DPPs sampling where items corresponding to large magnitudes are encouraged and the ones presenting strong correlations are suppressed. To incorporate the properties of DPPs with our goal of selecting informative and non-duplicate filters, we regard each filter as an item and formulate the subset selection problem as DPPs sampling.

We start from single layer pruning, e.g., ℓ-th layer, and then generalize it to multiple layers across the given network. The index ℓ is sometimes omitted if does not cause any confusion. Let $\boldsymbol{q}_i$ denote the informativeness factor of the i-th filter $\mathcal{F}_{\ell,i}$, and the marginal matrix $L\in\mathbb{R}^{N_{\ell+1}\times N_{\ell+1}}$ is calculated from $L_{ij}=\boldsymbol{q}_i S_{ij}\boldsymbol{q}_j$, where S_{ij} is the similarity between i-th and j-th filter. Therefore, the marginal matrix L contains all critical information needed to evaluate a subset, i.e., the individual filter importance and relationships among filters. Our goal is to find a subset Y containing filter index from a ground set $\mathcal{Y}=\{1,2,\ldots,N_{\ell+1}\}$ that maximizes the following objective function:

$$P(Y=\boldsymbol{y})=\frac{\det(L_{\boldsymbol{y}})}{\det(L+I)}, \tag{1}$$

where I is the identity matrix, $L_{\boldsymbol{y}}$ is the principal minor (submatrix) with rows and columns selected according to the indices in $\boldsymbol{y}$.

The determinant function $\det(\cdot)$ brings the property of pairwise repulsion. To see that, we take a subset of two filters i and j as an example. We have:

$$P(Y=\{i,j\})\propto \boldsymbol{q}_i^2\boldsymbol{q}_j^2(S_{ii}S_{jj}-S_{ij}^2). \tag{2}$$

If the filters i and j are the same, then $P(Y=\{i,j\})=0$ due to $S_{ij}=S_{ii}=S_{jj}$. Then identical filters are less likely to be selected simultaneously. If only one filter is preferred, then $P(Y=\{i\})\propto L_{i,i}=\boldsymbol{q}_i^2$, so that the filter with highest information will be selected. The general version also holds: $P(Y=\boldsymbol{y})\propto(\prod_{i\in\boldsymbol{y}}\boldsymbol{q}_i^2)\det(S_{\boldsymbol{y}})$. According to the first term $\prod_{i\in\boldsymbol{y}}\boldsymbol{q}_i^2$, the more informative filters are preferred, while similar filters are suppressed by the second term. With the integrated formulation, the diverse yet informative filters are expected to be selected.

As a comparison, most existing works [24], [42] select conv. filters using $\ell_{2,1}$-norm. Specifically, each filter is concatenated as a vector, then the filters with high value of $\ell_{2,1}$-norm will be selected. Although this kind of method could identify filters with large nonzero values, the dense filters with small values are also likely to be selected. To avoid this situation, Li *et al.* [1] select filters with large ℓ_1-norm so that the selected filters could

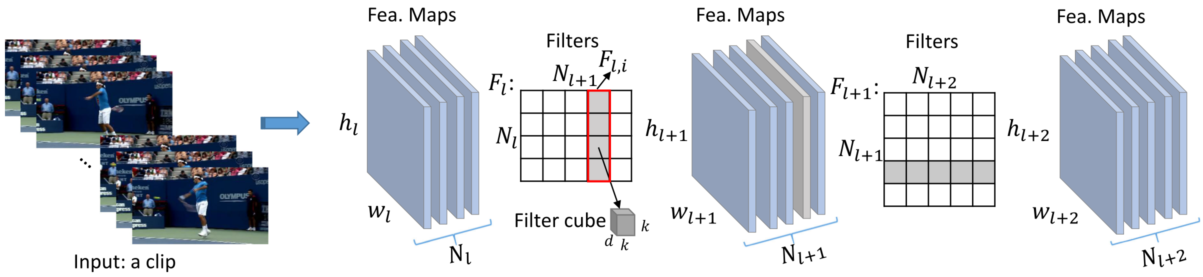


Fig. 2. Illustration of our proposed 3D filter pruning. Given a well-trained 3D ConvNets, the duplicate and less informative filters are pruned (highlighted by gray in $\mathcal{F}_\ell$). Consequently, the following feature maps can also be removed, so do the filter cubes corresponding to the removed feature maps.

be more sparse yet containing discriminative information. However, selecting filters only by this criterion may lead to duplicate filters. Our proposed method improves this by considering the relationships among filters together with the individual filter informativeness.

Optimization: As shown in many previous network compression studies [1], [5], [23], the accuracy drop is highly related with the compression ratio. High compression ratio usually leads to severe accuracy drop. Therefore, a controllable pruning ratio is necessary for adjusting to the requirements of different tasks. To this end, we resort to k-DPPs to optimize our objective function, where k is a manually fixed parameter denoting the size of the selected subset. Conditioning on fixed sampling size k, the k-DPPs sampling is formulated as:

$$P_k(Y = \boldsymbol{y}) = \frac{\det(L_{\boldsymbol{y}})}{\det(L + I)}, \text{s.t.} |Y| = k. \tag{3}$$

The k-DPP sampling is able to balance the trade-off between network size and network performance by tuning the parameter k for intended subset size. A larger k typically leads to a lower compression rate with smaller performance drop, and vice versa. For a layer with n_ℓ filters, the total number of pruning masks is $\mathcal{O}(2^{n_\ell})$ and an exhaustive search is therefore infeasible even for a small size network. For example, the greedy search for a fixed ratio of pruning, say $\frac{k}{n_\ell}$, $k < n_\ell$, still requires $\mathcal{O}(\mathcal{C}^k_{n_\ell})$. To address this problem, Anwar and Sung [43] propose to accelerate the sampling process by evaluating some random combinations. Such an ad-hoc strategy is highly dependent on these "random selected" combinations, and the best combination could be missed. In light of this, Li *et al.* [41] propose an efficient sampling optimization method specified for k-DPPs by constructing coresets for the ground set of items. By leveraging the fast k-DPPs, the complexity of our proposed sampling is reduce to $\mathcal{O}(n_\ell k^3)$. In our experiments, this sampling cost is negligible compared to the cost of training.

B. Constructing DPPs Marginal Matrix L

Recall that the marginal matrix L is constructed from the individual filter importance score $\boldsymbol{q}_i$ and the pairwise similarities S_{ij}. For the ℓ-th layer of a well-trained 3D ConvNets, the 5-dimensional tensor $W^\ell \in \mathbb{R}^{n_{\ell+1} \times n_\ell \times k \times k \times d}$ can be decomposed to $n_{\ell+1}$ filters $\{\mathcal{F}_{\ell,i} \in \mathbb{R}^{n_\ell \times k \times k \times d}\}_{i=1}^{n_{\ell+1}}$. Each filter is associated with a scalar $\boldsymbol{q}_i$ denoting the informativeness score and a vector Φ_i denoting the filter representation.

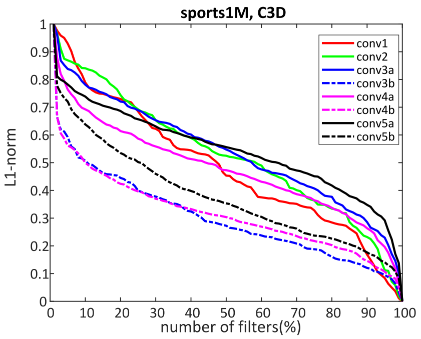


Fig. 3. Visualization the distribution of importance score $\boldsymbol{q}_i$ of filters in all convolutional layers of C3D trained on Sports1M.

Informativeness score $\boldsymbol{q}_i$: In the filter-level pruning, many existing works use $\ell_{2,1}$-norm as a regularizer [23], [24], [42] and obtain promising results. Recent work [1] shows that there is no significant difference between the ℓ_2-norm and the ℓ_1-norm for filter selection since the important filters tends to have large values under both measures. Following [1], we measure the relative informativeness $\boldsymbol{q}_i$ of a filter in ℓ-th layer by ℓ_1-norm $\|\mathcal{F}_{\ell,i}\|_1$, and obtain

$$\boldsymbol{q} = \left[\|\mathcal{F}_{\ell,1}\|_1, \|\mathcal{F}_{\ell,2}\|_1, \ldots, \|\mathcal{F}_{\ell,n_{\ell+1}}\|_1\right]^\top. \tag{4}$$

In Fig. 3, we show the distribution of filters' ℓ_1-norm for each convolutional layer in a C3D network well-trained on Sports1M dataset. The sum of all absolute weights of a filter provides a clue for the magnitude of the output feature maps. Compared with other filters in that layer, filters with larger weights are likely to produce feature maps with stronger activations. In this sense, our filter pruning method which preserves filters with large weights, although being data-independent, is able to capture the information that is obtained from the activation-based feature maps pruning methods.

Filter representation Φ_i: Motivated by the Maximum Activation of Convolutions (MAC) feature representations [11], which generate compact representations from feature maps of convolutional layers. The 3-dimensional feature maps consist of multiple channels, and MAC is obtained by taking the maximum value of each channel. Similar to feature maps, each filter can be

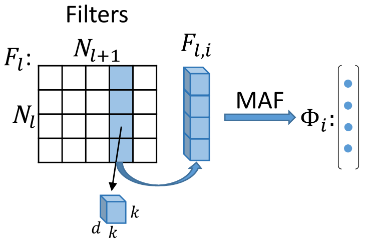


Fig. 4. Maximum Abs of Filters (MAF) representation. The filter cube with the maximal absolute values are concatenated as the compact filter representation Φ_i.

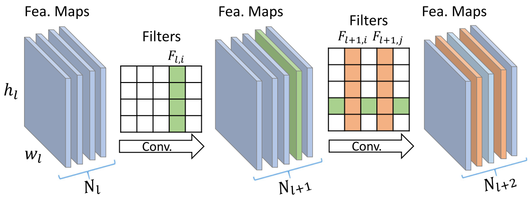


Fig. 5. Filter pruning across multiple layers. The filters pruned in the ℓ-th layer (green by column) will affect the number of feature maps generated by them and also the size of filters in $(\ell + 1)$-th layer (green by row).

seen as a pillar piled up by 3-dimensional filter cube (see Fig. 4). Every convolution operation with a filter cube will generate a numerical value in the corresponding feature map of next layer through sum of multiplications. Hence, only large weights have a chance to dominate the activations of the following feature maps. Based on this observation, we concatenate the weights with maximal absolute values of a filter cube, obtaining the compact filter representation $\Phi_i \in \mathbb{R}^{n_l}$. Then it is post-processed by ℓ_2 normalization. We term this kind of representations as MAF (Maximum Absolute value of Filters). The cosine similarity is utilized to measure pairwise filters:

$$S_{ij} = \Phi_i^\top \Phi_j. \tag{5}$$

C. Multiple Layers Pruning

Inspired by [1], [44], we prune a given network layer by layer. For each layer, we apply the k-DPPs sampling to all filters. Except for the selected k filters, the remaining will be pruned and the corresponding feature maps generated by those filters will consequently be removed, so do the filter cubes associated with those feature maps in the next convolutional layer (see Fig. 5). Subsequently, the filters pruned in the ℓ-th layer can affect the number of feature maps generated by them and also the size of filters in $(\ell + 1)$-th layer which operates convolution on previously removed feature maps. Besides, the filters in $(\ell + 1)$-th layer will simultaneously apply pruning on themselves.

Since the learning capability of a network is determined by its architecture and the number of effective learnable parameters, removing a small number of unimportant filters out will not affect the network accuracy significantly. However, a heavy pruning will inevitably degrade the accuracy. To compensate for the accuracy loss caused by pruning, we apply a retraining step after all layers have been pruned. Algorithm 1 summarizes our training process.

Algorithm 1: 3D Pruning for a Well-Trained Network

```
Input: A well-trained network, and the pruning ratio $\rho \in (0, 1)$.
Output: Pruned network, the pruned neural network object including its weights.
for $\ell \leftarrow 1$ to $L$ do
    Informativeness for each filter: $\boldsymbol{q}_i \leftarrow \|\mathcal{F}_{\ell,i}\|_1$
    Mutual similarity: $S_{ij} \leftarrow \Phi_i^\top \Phi_j$
    Build Objective from learned variable $\boldsymbol{q}_i$ and $S_{ij}$
    Optimize objective function using k-DPPs [30], get subset $Y_\ell$
end for
Parameterize the network with $\{Y_\ell\}_{\ell=1}^L$
```

IV. Experiments

To evaluate our proposed 3D ConvNets pruning method, we conduct experiments on the extensively used C3D network [13] and P3D network [14] for video classification on Sports-1 M dataset [15]. Our proposed method can also be flexibly extended to 2D network pruning, which is evaluated on the popular VGG-16 network [18] and ResNet-18/50/101 [19] for image classification with datesets including ImageNet [17] and Cifar10 [16]. All experiments are performed on a 12 G Tesla K40 GPU using PyTorch. The statistical model information and classification accuracy of the original model are shown in Table I. The number of parameters is mainly dependent on the networks (minor difference on the last classification layer), while the number of Flops is related to both the width and depth of networks and also the input size. Although the same baseline network is used, the different input sizes of ImageNet ($224 \times 224 \times 3$) and Cifar10 ($32 \times 32 \times 3$) lead to the huge difference in the number of Flops and also the model running time.

Since we did not observe any work specifically designed for 3D ConvNets pruning, we compare our 3D pruning method with some variations that are highly related to ours. Remember that we apply DPPs sampling to select the informative and non-duplicate filters, the informative score is measured by the ℓ_1-norm of each filter. We assume filters with larger ℓ_1-norms contain more information. Considering our settings, the controlled experiments should be able to verify (1) does large ℓ_1-norm indicate high informativeness; (2) whether the non-duplicate factor is useful or not? Based on this, we have made the following comparisons:

- **ℓ_1-Max:** To evaluate the effectiveness of DPPs sampling, especially the non-duplicate ingredient, we compare with the setting that large ℓ_1-norm is the only criterion for selecting filters. The large ℓ_1-norm policy is also applied in [1] for 2D filter pruning. Here, ℓ_1-Max denotes the filters with larger ℓ_1-norms are selected.
- **Random:** In [43], Sajid and Sung evaluate some random combinations of 2D filters, from which the combination that causes the least degradation to the network accuracy is selected. We compare our method with this random sampling strategy to validate our assumption that large ℓ_1-norm indicates high importance.

TABLE I
STATISTICS OF BASELINE MODELS

Network	C3D	P3D	VGG16		ResNet-18	ResNet-50		ResNet-101
Dataset	Sports1M		Cifar10	ImageNet	Cifar10	Cifar10	ImageNet	ImageNet
#Params	8.0e+07	6.5e+07	1.5e+07		1.1e+07	2.6e+07		4.5e+07
#Flops	5.6e+09	1.2e+10	0.3e+09	1.5e+10	1.3e+09	3.2e+09	4.1e+09	7.8e+09
Top-1 Acc (%)	57.9	63.4	91.6	68.7	93.5	93.6	72.8	77.4
Top-5 Acc (%)	82.6	84.4	-	87.2	-	-	91.1	93.6

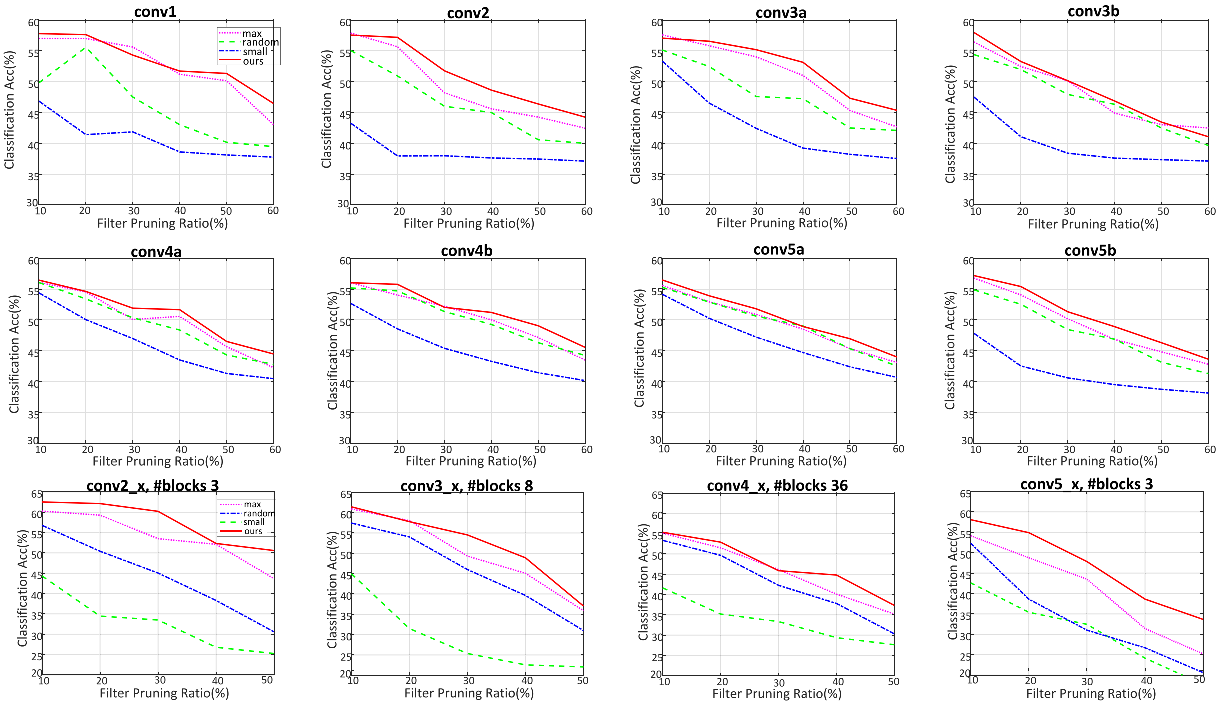


Fig. 6. C3D (**Top**) and P3D (**Bottom**) network on Sports1M: single layer performance analysis under different pruning ratios (w/o retraining).

- ℓ_1**-Small:** For a comprehensive comparison, we also show the performance of pruned model on the setting that the filters with small ℓ_1-norms are selected.

To save space, we abbreviate the above descriptions on comparisons as ℓ_1-Max, Random and ℓ_1-Small in the following figures and tables.

A. 3D ConvNets Pruning

Sports1M is a large video classification benchmark, consisting of about 1.1 million sports videos labeled as 487 sports categories. As sports1M has many long videos, we follow [13] to randomly extract five 2-second long clips from every video, then sample 16 key frames from every clip. Clips are resized to have a frame size of 128 $\times$ 171, and then centrally cropped into 112 $\times$ 112. We report the video classification results. The details of the backbone networks are shown below:

1) **C3D:** C3D [13] is a 10-layer single-branch 3D convolutional neural network, with 8 convolutional layers. It is well-designed for capturing spatiotemproal features, thus has been widely used in video related topics, such as action recognition [45]–[47].

2) **P3D199:** In [14], multiple P3D variants are provided, we choose the complete design of P3D, dubbed P3D199, which is designed based on the 152-layer ResNet [19]. It contains four stages of residual blocks, the number of blocks in each stage is 3-8-36-3, respectively. Each block consists of four conv. layers, two vanilla 3D conv. layers located at the beginning and the end, between which are a spatial conv. layer and a temporal conv. layer. The pair of the spatial and temporal layers are integrated to construct the DPPs marginal matrix L for filters selection.

For the two backbone networks, we use the well-trained network on the Sports1M train split as our base network, and test the pruned model on the test split.

Single Layer Pruning: In this part, we compare our proposed 3D filter pruning method with the above mentioned baselines: ℓ_1-Max, ℓ_1-Random and ℓ_1-Small in a layer-by-layer manner. Fig. 6 shows the curves of classification accuracy on Sports1M changing with filter pruning ratios for all convolutional layers of C3D network and P3D199 network. As expected, the accuracy degrades as pruning ratio increasing. Our method and ℓ_1-Max performs consistently better than ℓ_1-Random and ℓ_1-Small, which suggests that filters with larger (abs) values

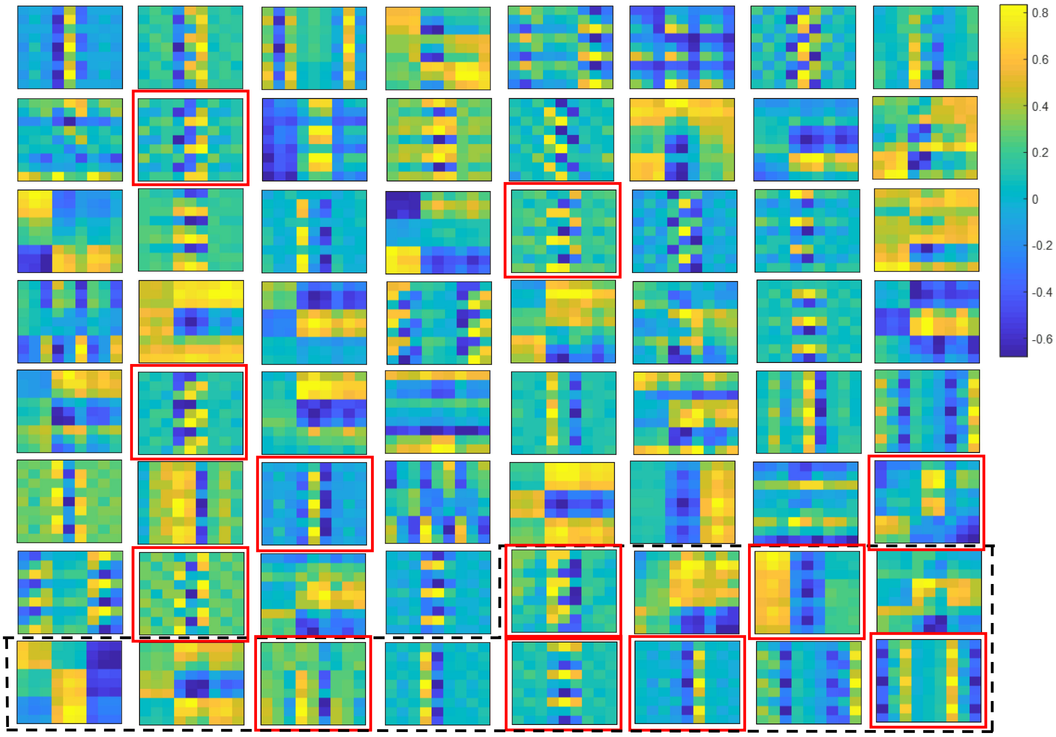


Fig. 7. Visualization of filters in conv1 of C3D network trained on Sports1M, with about 20% (12/64) pruning ratio. Filters are ranked by its importance score $\boldsymbol{q}_i$ in a descending order from top left to bottom right. Filters with red boxes are the ones pruned by our method, the last 12 filters circled by black dashed lines are the ones pruned according to criteria in [1], i.e., pruned the ones with least ℓ_1-norm.

have better representative power. One can also observe that our 3D pruning method generally outperforms ℓ_1-Max, the gaps between those two methods are owing to the effective DPPs sampling catering for both representative and dicriminative filters. Interestingly, ℓ_1-Random sometimes leads to better accuracy than ℓ_1-Max. As we conjectured in Sect. 3.1, this may be caused by the duplicates among filters with large ℓ_1-norm, i.e., the filters selected by ℓ_1-Max strategy are not diverse enough.

To further validate this conjecture, we visualize the 3D filters in the first convolutional layer of C3D network in Fig. 7. There are 64 filters in $\mathbb{R}^{3\times3\times3\times3}$ in total, and those filters are ranked by its ℓ_1-norm in a descending order. If the ℓ_1-Max strategy is applied to prune, say with ratio $= \frac{12}{64}(18.75\%)$, then the top 52 filters will all be selected (filters except the ones circled by black dashed lines). It is obvious that some selected filters are highly similar to each other, and some pruned filters may contain unique information/knowledge. As a result, some functionally diverse filters are probably missed. To avoid such a circumstance, our DPPs-based method will select filters that are more representative while considering their similarities (filters except the ones circled by red boxes), thus the pruned results can be marked improved over ℓ_1-Max.

Multiple Layers Pruning: As introduced in Sect. 3.3, we prune the whole network layer by layer. That is, after filters being pruned in each layer, we combine all convolutional layers and trim the rest filters according to the strategy showed in Fig. 5. Then we fine-tune the pruned model with a decreasing learning rate starting from 0.0015 for 6 epochs on C3D and 0.0005 for 8 epochs on P3D199. The results are shown in Table II. We achieve about 75% FLOPs reduction with only 1.94% Top-5 performance drop using C3D network, and 1.69% acc. drop using P3D network. The top-1 performance is still satisfactory at about 50% FLOPs reduction for both networks with less than 1% acc. drop. The P3D199, which is constructed based on Resnet-152, performs slightly better than C3D. One possible reason is that the residual design could preserve the information from previous layers that is pruned at current layer. It is foreseeable that the performances of ℓ_1-Random and ℓ_1-Small on both networks are undesirable, which again reflects the rationality of using ℓ_1-norm as the indicator of importance for filters. The superior performance of our proposed method over ℓ_1-Max demonstrates the effectiveness of using DPPs to select the representative and yet diverse filters.

Comparison to State-Of-The-Art (SOTA): There are mainly two direction in network pruning, weights based pruning and activation based pruning. The main difference of the two network pruning directions is that the former is data independent, so that given a well-trained network, we are able to directly remove the redundant weights. However, the activation based pruning compresses a network by measuring its activations, which are highly dependent on the inputs. And the pruned network needs to be updated once unseen samples are available. Thus the generalization ability of weights based pruning is superior to that of activation based, and the majority studies recently published are on weights pruning.

TABLE II

COMPARISON OF SEVERAL BASELINES AND OUR PROPOSED ON SPORTS1M DATASET. **TOP**: C3D. **BOTTOM**: P3D199

Pruning Ratio	Saved FLOPs	Max		Random		Small		Ours	
		Top-1(%)↓	Top-5(%)↓	Top-1(%)↓	Top-5(%)↓	Top-1(%)↓	Top-5(%)↓	Top-1(%)↓	Top-5(%)↓
10%	18.51%	0.22	0.13	0.42	0.26	1.21	0.47	0.1	0
20%	35.59%	0.78	0.35	1.7	0.74	2.59	0.79	0.36	0.11
30%	50.53%	1.41	0.91	2.52	1.62	4.07	2.63	0.93	0.44
40%	63.52%	2.76	1.84	4.94	2.77	9.73	6.57	1.76	0.89
50%	74.73%	5.11	3.32	8.4	4.39	14.82	10.83	3.79	1.94

Pruning Ratio	Saved FLOPs	Max		Random		Small		Ours	
		Top-1(%)↓	Top-5(%)↓	Top-1(%)↓	Top-5(%)↓	Top-1(%)↓	Top-5(%)↓	Top-1(%)↓	Top-5(%)↓
10%	18.48%	0.17	0.15	0.41	0.19	1.36	0.58	0.04	0
20%	35.08%	0.65	0.26	1.18	0.57	2.4	0.72	0.27	0.08
30%	49.79%	1.4	0.73	1.79	1.43	4.14	2.31	0.81	0.3
40%	62.61%	2.53	1.55	4.28	2.82	9.26	6.45	1.61	0.67
50%	73.56%	4.76	3.17	7.21	4.35	13.79	10.26	3.52	1.69

TABLE III

COMPARISON OF SOTA AND OUR PROPOSED METHOD ON SPORTS1M DATASET. **TOP**: C3D. **BOTTOM**: P3D199

Pruning Ratio	Saved FLOPs	Xavier *et al.* [33]		Luo *et al.* [35]		ℓ_1-act.		Pavlo *et al.* [22]		Ours	
		Top-1(%) ↓	Top-5(%) ↓	Top-1(%) ↓	Top-5(%) ↓	Top-1(%) ↓	Top-5(%) ↓	Top-1(%) ↓	Top-5(%) ↓	Top-1(%) ↓	Top-5(%) ↓
10%	18.51%	0.52	0.33	0.13	0.02	1.32	0.51	0.11	0.04	0.10	0
20%	35.59%	1.35	0.74	0.40	0.15	2.24	0.86	0.40	0.14	0.36	0.11
30%	50.53%	1.72	1.15	0.90	0.45	3.97	2.54	0.97	0.54	0.93	0.44
40%	63.52%	3.35	2.19	1.91	0.93	7.39	5.44	1.59	0.88	1.76	0.89
50%	74.73%	5.28	3.96	4.02	2.12	10.28	7.65	4.00	2.03	3.79	1.94

Pruning Ratio	Saved FLOPs	Xavier *et al.* [33]		Luo *et al.* [35]		ℓ_1-act.		Pavlo *et al.* [22]		Ours	
		Top-1(%) ↓	Top-5(%) ↓	Top-1(%) ↓	Top-5(%) ↓	Top-1(%) ↓	Top-5(%) ↓	Top-1(%) ↓	Top-5(%) ↓p	Top-1(%) ↓	Top-5(%) ↓
10%	18.48%	0.47	0.30	0.07	0.02	1.41	0.63	0.05	0.01	0.04	0
20%	35.08%	1.22	0.83	0.32	0.14	2.32	0.91	0.26	0.11	0.27	0.08
30%	49.79%	1.85	1.29	0.97	0.40	4.10	2.67	0.91	0.57	0.81	0.30
40%	62.61%	3.46	2.27	1.88	0.75	9.45	6.14	1.60	0.62	1.61	0.67
50%	73.56%	6.08	4.19	3.96	1.53	10.47	8.65	3.60	1.47	3.52	1.69

We compare our proposed method with both activation-based and weight-based methods [21], [32], [34], [48]. The first two compared methods, [34], [48] and [32], are representative research directions for activation based pruning, we use the released code from [34] and implement [32] according to their paper. We also design a baseline ℓ_1-norm of a activations/feature maps, denoted as ℓ_1-act. The basic idea of the third method [21] is to evaluate a filter by the performance drop after removing it from the network. In [21] a few variations are proposed, we use the first-order importance estimate as it is most frequently used in their experiments.

The results are shown in Table III, from which we can see that ℓ_1-act. performs remarkably worse than the other methods. Considering that ℓ_1-norm of filters is the prevailing measurement for weights based methods, the observation suggests that the importance of a feature map is not closely related to the ℓ_1-norm as to filters. Although better than ℓ_1-act., [32] is still inferior to our method by a large margin. ThiNet [34] is comparative to our method, however, its computational complexity is $\mathbb{O}(C_{n_\ell}^k)$, while that of our method is $\mathbb{O}(n_\ell k^3)$. The weight-based method [21] performs slightly worse than ours in most cases, the possible reason is that [21] is under an *i.i.d.* assumption of all filters, and the importance of each filter is evaluated individually based on its influence to the ultimate classification accuracy. However, we argue that the relations between filters matter. A large set of informative but duplicate filters is sub-optimal and potentially subject to further pruning into a small set of representative ones, or being improved by the same number of informative yet diverse filters, which makes the key difference between our method and [21].

Ablation Study: To evaluate the effectiveness of the proposed compact filter representation Maximum Abs. of Filters (MAF), we conduct experiments using the naive vectorized filter representation for an ablation study. As our method is weights based pruning, which is closely related to the activation based pruning, we thus compare with several representative works on activation based methods.

We compare the proposed MAF with the naive vectorized filter representation (abbreviated as Vec.) while applying to the filter pruning for the whole network, table IV shows the results. From the table we can see that the MAF is slightly superior to Vec. filter on 3D network pruning, and is comparative to Vec. filter on 2D network pruning. One possible reason is that the redundancy in 3D network is more serious than that in 2D network. Besides, the proposed MAF is much more compact than the vectorized filter representation, thus consumes less memory for storage and short time for similarity calculation.

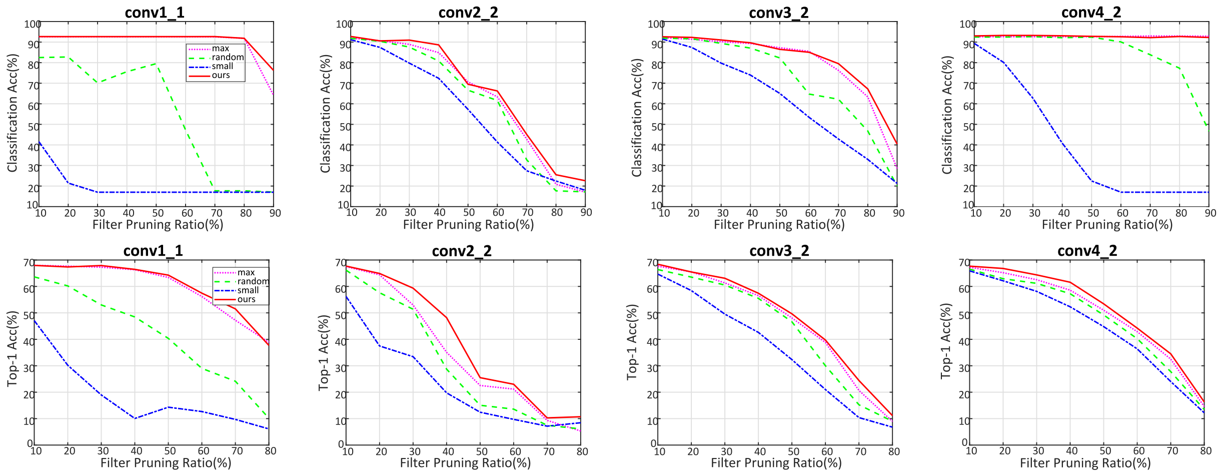


Fig. 8. VGG-16 network on Cifar10 (**Top**) and ImageNet (**Bottom**): Single layer performance analysis with different pruning ratios (w/o retraining).

TABLE IV
Comparisons of the Proposed MAF to the Naive Vectorized Filter Representation (Top-1 Acc Drop (%), With Fine-Tuning)

Pruning Ratio	C3D/Sports1M MAF / Vec.	P3D/Sports1M MAF / Vec.	VGG16/Cifar10 MAF / Vec.	VGG16/ImageNet MAF / Vec.
10%	0.1 / 0.13	0.04 / 0.04	0.0 / 0.0	0.80 / 0.71
20%	0.36 / 0.40	0.27 / 0.26	0.06 /0.04	1.24 / 1.25
30%	0.93 / 0.93	0.81 / 0.83	0.14 / 0.15	1.97 / 1.94
40%	1.76/ 1.81	1.61 / 1.70	0.29 / 0.26	2.63 / 2.70
50%	3.76 / 3.78	3.52 / 3.55	0.41 / 0.40	3.83 / 3.76

TABLE V
Comparisons of Pruning Results on Cifar10 in Terms of Various Pruning Ratios (PR)

Network	Method	PR	FLOPs ↓	Acc.(%) ↓
VGG16	Li *et al.* [1]	-	34.2%	0.13
	Huang *et al.* [49]	-	55.2%	1.1
	Hu *et al.* [50]	-	56.2%	0.24
	Ours	10%	18.29%	0.0
		20%	35.27%	0.06
		30%	50.45%	0.14
		40%	63.45%	0.29
		50%	74.86%	0.41
		60%	83.58%	0.97
		70%	90.65%	1.72
		80%	95.79%	2.26
ResNet-18	Li *et al.* [1]	7.2%	7.5%	1.06
	Huang *et al.* [49]	-	64.5%	1.9
	Pavlo *et al.* [22]	-	25.0%	0.44
	Yang *et al.* [21]	40%	54.0%	1.76
	Ours	10%	11.25%	0.0
		20%	22.93%	0.05
		30%	33.96%	0.11
		40%	43.71%	0.24
		50%	54.06%	0.36
ResNet-50	Li *et al.* [1]	13.7%	27.6%	1.73
	Hu *et al.* [50]	20%	56.2%	0.49
	Pavlo *et al.* [22]	-	25.0%	0.32
	Yang *et al.* [21]	40%	52.6%	0.1
	Lin *et al.* [26]	-	48.5%	0.52
	Xavier *et al.* [33]	-	61.5%	0.40
	Ours	10%	13.13%	0.0
		20%	23.77%	0.04
		30%	36.21%	0.09
		40%	45.26%	0.20
		50%	55.14%	0.31

B. 2D ConvNets Pruning

For a fair comparison with existing filter pruning method, we extend our proposed 3D pruning method to 2D ConvNets. The importance score $\boldsymbol{q}_i$ is calculated similar to 3D ConvNets using ℓ_1-norm. The filter representation Φ_i is now obtained by taking the max (abs) value of 2-dimensional kernels rather than the filter cube in 3D ConvNets. Cosine similarity is applied to measure the relationships between filters. We evaluate the extensively used VGG-16 network and ResNet-18/50/101 on Cifar10 and ImageNet.

Single Layer Pruning: Fig. 8 shows several representative layers' pruning results of VGG-16 on Cifar10 and ImageNet. On both datasets, our proposed method and ℓ_1-Max perform marginally better than ℓ_1-Random and ℓ_1-Small, which is consist with our observation on 3D ConvNets. For VGG-16 on Cifar10, the results vary a lot in different layers. Specifically, the filter pruning on last layers does not influence the classification results due to the small size of input image and the relative very deep network architecture. For VGG-16 on ImageNet, the accuracy drops dramatically with pruning ratio increasing, which suggests the majority of filters are critical for a satisfactory performance.

Comparison to SOTA w.r.t. Multiple Layers Pruning: The networks are pruned layer by layer, then they afine-tuned to boost the performance. For VGG-16 onCifar10/ImageNet, it is fined-tuned for 10/23 epochs with a fixed learning re rate 0.001. For ResNet-18/-50 on Cifar10, it is fine-tuned for 10/12 epochs with a decreasing learning rate from 0.01. For ResNet-50/-101 on ImageNet, it is fine-tuned for 25/30 epochs with a decreasing learning rate from 0.01. Table V and Table VI show comparisons of various networks on Cifar10 and ImageNet. For VGG-16 on Cifar10, 34.2% FLOPs reduction is achieved with 0.13% accuracy drop in [1] and 55.2% FLOPs reduction with 1.1% accuracy drop in [49]. Our proposed method outperforms [1], [49]

TABLE VI
COMPARISONS OF PRUNING RESULTS ON IMAGENET IN TERMS OF VARIOUS PRUNING RATIOS (PR)

Network	Method	PR / FLOPs↓	Top-1(%)↓	Top-5(%)↓
VGG16	Pavlo [51]	- / 72.9%	5.12	5.44
	Li [1]	- / 15.5%	0.76	0.51
	Luo [35]	- / 68.39%	2.98	1.5
	Lin [52]	- / 58.71%	1.52	0.65
	Ours	10% / 18.83%	0.8	0.44
		20% / 35.52%	1.24	0.7
		30% / 50.71%	1.97	1.12
		40% / 63.57%	2.63	1.78
		50% / 74.90%	3.83	2.42
ResNet-50	Li [1]	30% / 36.78%	2.56	1.49
	Luo [35]	30% / 36.78%	0.84	0.47
	Lin [52]	30% / 36.78%	3.09	1.63
	Yang [21]	40% / 53.5%	1.32	0.55
	Lin [26]	16.8% / 43.02%	4.20	1.93
	Lin [26]	42.5% / 61.36%	6.27	3.12
	Zhao [27]	40% / -	2.30	1.70
	Ours	10% / 13.45%	0.07	0.02
		20% / 24.61%	0.37	0.16
		30% / 36.78%	0.94	0.40
		40% / 45.73%	1.73	1.12
		50% / 55.82%	2.61	1.39
ResNet-101	Li [1]	30% / 39.62%	0.33	0.24
	Li [1]	50% / 65.14%	1.24	0.71
	Pavlo [22]	30% / 39.74%	0.02	-
	Yang [21]	30% / 42.2%	0.05	0.0
	Ours	10% / 14.19%	0.01	0.0
		20% / 27.30%	0.06	0.02
		30% / 41.52%	0.11	0.07
		40% / 53.67%	0.25	0.10
		50% / 72.49%	0.52	0.21

by 0.06% accuracy drop at 35.27% FLOPs saving and 0.29% accuracy drop at 63.45% FLOPs saving, respectively. [50] and our method performs similarly with 0.24% acc. drop at 56.2% FLOPs saving versus 0.29% acc. drop at 63.45% FLOPs saving. For the ResNet, the pruning ratio of [1] is clearly lower than our method, while [49] and [50] are promising on saving FLOPs, but suffer from relatively large accuracy drop. For those recent work from CVPR'19 like [20], [21], [25], [32], they usually save less FLOPs under the same rate of accuracy drop, except the case of [20] on ResNet-50, where the accuracy drop is slightly better by 0.1%.

We can observe similar phenomenons on larger datasets and deeper networks. For VGG-16 on ImageNet, although [52] is slightly better than ours, the method requires extra prior knowledge about the importance of filters which leads to an unfair comparison. The proposed method performs favourably against [1] and [51], is comparable to [34] with 2.63% acc. drop at 63.57% FLOPs saving verses 2.98% acc. drop at 68.39% FLOPs saving. Note that [1] is only based on ℓ_1-norm for selecting filters, i.e., the baseline of ℓ_1-Max, the inferior performance on 2D ConvNet pruning again verifies the effectiveness of our proposed DPPs-based method. For ResNet-10/101 on ImageNet, [1], [34] and [52] demonstrate lower performances than our method if we consider pruning ratio, FLOPs and accuracy drop in a whole. For those recent work from CVPR'19, [21] and [20] demonstrate promising performances on par with our method, while [25] and [26] suffer from significant accuracy drop, i.e., 6.27% and 2.30%, respectively.

V. CONCLUSION

In this paper, we have proposed a new filter pruning method for accelerating 3D ConvNets. Filter pruning for 3D ConvNets is critical to deal with large volume of video data and it is feasible owing to the considerable redundancy in consecutive video frames. To our best knowledge, the proposed method is a pioneer work to explore filter pruning in 3D ConvNets. We leverage DPPs, which integrate filter importance and the similarities between filters into a unified model, for selecting informative yet diverse 3D filters. Additionally, a novel filter representation MAF is introduced to measure the similarities between filters. Our proposed method achieves 50.53% FLOPs reduction with only 0.93% drop in accuracy for C3D network on Sports1M dataset. We also extend our method to 2D filter pruning and achieve performance comparable with state-of-the-art filter pruning methods for 2D ConvNets.

ACKNOWLEDGMENT

This research was carried out at the Rapid-Rich Object Search (ROSE) Lab at the Nanyang Technological University, Singapore.

REFERENCES

[1] H. Li, A. Kadav, I. Durdanovic, H. Samet, and H. P. Graf, "Pruning filters for efficient convnets," in *Proc. Int. Conf. Learn. Representations*, 2017, pp. 1–13.

[2] D. Wang, P. Cui, M. Ou, and W. Zhu, "Learning compact hash codes for multimodal representations using orthogonal deep structure," *IEEE Trans. Multimedia*, vol. 17, no. 9, pp. 1404–1416, Sep. 2015.

[3] W. Hong, J. Yuan, and S. D. Bhattacharjee, "Fried binary embedding for high-dimensional visual features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6221–6229.

[4] W. Hong, Z. Wang, M. Yang, and J. Yuan, "Conditional generative adversarial network for structured domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1335–1344.

[5] Y. He, X. Zhang, and J. Sun, "Channel pruning for accelerating very deep neural networks," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 1389–1397.

[6] J. Hou, X. Wu, Y. Sun, and Y. Jia, "Content-attention representation by factorized action-scene network for action recognition," *IEEE Trans. Multimedia*, vol. 20, no. 6, pp. 1537–1547, Jun. 2018.

[7] P. Wang, W. Li, Z. Gao, C. Tang, and P. O. Ogunbona, "Depth pooling based large-scale 3-d action recognition with convolutional neural networks," *IEEE Trans. Multimedia*, vol. 20, no. 5, pp. 1051–1061, May 2018.

[8] J. Wang, W. Wang, and W. Gao, "Multiscale deep alternative neural network for large-scale video classification," *IEEE Trans. Multimedia*, vol. 20, no. 10, pp. 2578–2592, Oct. 2018.

[9] Y.-G. Jiang *et al.*, "Modeling multimodal clues in a hybrid deep learning framework for video classification," *IEEE Trans. Multimedia*, vol. 20, no. 11, pp. 3137–3147, Nov. 2018.

[10] T. Yu, Z. Wang, and J. Yuan, "Compressive quantization for fast object instance search in videos," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 726–735.

[11] G. Tolias, R. Sicre, and H. Jégou, "Particular object retrieval with integral max-pooling of CNN activations," in *Proc. Int. Conf. Learn. Representations*, 2016, pp. 1–12.

[12] C. Li, S. Jegelka, and S. Sra, "Fast DPP sampling for nyström with application to kernel methods," in *Proc. Int. Conf. Int. Conf. Mach. Learn.*, 2016, pp. 2061–2070.

[13] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.

[14] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3d residual networks," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 5533–5541.

[15] A. Karpathy *et al.*, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 1725–1732.

[16] A. Krizhevsky and G. Hinton, "Learning multiple layers of features from tiny images," Univ. Toronto, Tech. Rep., vol. 1, no. 4, pp. 1–60, 2009.

[17] J. Deng *et al.*, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

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

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[20] Y. He, P. Liu, Z. Wang, Z. Hu, and Y. Yang, "Filter pruning via geometric median for deep convolutional neural networks acceleration," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4340–4349.

[21] P. Molchanov, A. Mallya, S. Tyree, I. Frosio, and J. Kautz, "Importance estimation for neural network pruning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 11264–11272.

[22] A. G. Howard *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv preprint arXiv:1704.04861*.

[23] V. Lebedev and V. Lempitsky, "Fast convnets using group-wise brain damage," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2554–2564.

[24] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li, "Learning structured sparsity in deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 2082–2090.

[25] S. Lin *et al.*, "Towards optimal structured CNN pruning via generative adversarial learning," in *Proc. IEEE Conf. Comput. Vision Pattern Recognit.*, 2019, pp. 2790–2799.

[26] C. Zhao *et al.*, "Variational convolutional neural network pruning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2780–2789.

[27] Z. Liu *et al.*, "Learning efficient convolutional networks through network slimming," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 2736–2744.

[28] S. Lin, R. Ji, Y. Li, C. Deng, and X. Li, "Toward compact convnets via structure-sparsity regularized filter pruning," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published.

[29] Y. Xu, Y. Wang, A. Zhou, W. Lin, and H. Xiong, "Deep neural network compression with single and multiple level quantization," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 4335–4342.

[30] J. Ye, X. Lu, Z. Lin, and J. Z. Wang, "Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers," in *Proc. Int. Conf. Learn. Representations*, 2018, pp. 1–11.

[31] A. Polyak and L. Wolf, "Channel-level acceleration of deep face representations," *IEEE Access*, vol. 3, pp. 2163–2175, 2015.

[32] X. Suau, L. Zappella, V. Palakkode, and N. Apostoloff, "Principal filter analysis for guided network compression," 2018, *arXiv:1807.10585*.

[33] Y. Li *et al.*, "Exploiting kernel sparsity and entropy for interpretable CNN compression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2800–2809.

[34] J.-H. Luo, J. Wu, and W. Lin, "Thinet: A filter level pruning method for deep neural network compression," in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 5058–5066.

[35] Y. LeCun, J. S. Denker, and S. A. Solla, "Optimal brain damage," in *Proc. Adv. Neural Inf. Process. Syst. 2*, 1990, pp. 598–605.

[36] B. Hassibi and D. G. Stork, "Second order derivatives for network pruning: Optimal brain surgeon," in *Proc. Adv. Neural Inf. Process. Syst. 5*, 1993, pp. 164–171.

[37] T. N. Sainath, B. Kingsbury, V. Sindhwani, E. Arisoy, and B. Ramabhadran, "Low-rank matrix factorization for deep neural network training with high-dimensional output targets," in *Proc. Int. Conf. Acoust., Speech Signal Process.*, 2013, pp. 6655–6659.

[38] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1135–1143.

[39] Z. Mariet and S. Sra, "Diversity networks: Neural network compression using determinantal point processes," in *Proc. Int. Conf. Learn. Representations*, 2016, pp. 1–13.

[40] H. Song, H. Mao, and W. J. D. William, "Deep compression: Compressing deep neural network with pruning, trained quantization and Huffman coding," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–13.

[41] C. Li, S. Jegelka, and S. Sra, "Efficient sampling for k-determinantal point processes," *J. Mach. Learn. Res.*, vol. 51, pp. 1328–1337, 2016.

[42] H. Zhou, J. M. Alvarez, and F. Porikli, "Less is more: Towards compact CNNS," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 662–677.

[43] S. Anwar and W. Sung, "Compact deep convolutional neural networks with coarse pruning," 2016, *arXiv:1610.09639*.

[44] X. Zhang, J. Zou, K. He, and J. Sun, "Accelerating very deep convolutional networks for classification and detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 1943–1955, Oct. 2016.

[45] C. Bak, A. Kocak, E. Erdem, and A. Erdem, "Spatio-temporal saliency networks for dynamic saliency prediction," *IEEE Trans. Multimedia*, vol. 20, no. 7, pp. 1688–1698, Jul. 2018.

[46] N. Takahashi, M. Gygli, and L. Van Gool, "Aenet: Learning deep audio features for video analysis," *IEEE Trans. Multimedia*, vol. 20, no. 3, pp. 513–524, Mar. 2018.

[47] L. Gao, Z. Guo, H. Zhang, X. Xu, and H. T. Shen, "Video captioning with attention-based LSTM and semantic consistency," *IEEE Trans. Multimedia*, vol. 19, no. 9, pp. 2045–2055, Sep. 2017.

[48] J.-H. Luo *et al.*, "Thinet: Pruning CNN filters for a thinner net," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 10, pp. 2525–2538, Oct. 2018.

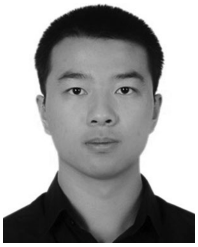
[49] Q. Huang, K. Zhou, S. You, and U. Neumann, "Learning to prune filters in convolutional neural networks," in *Proc. Int. Workshop Appl. Comput. Vis.*, 2018, pp. 709–718.

[50] Y. Hu, S. Sun, J. Li, X. Wang, and Q. Gu, "A novel channel pruning method for deep neural network compression," 2018, *arXiv:1805.11394*.

[51] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient transfer learning," in *Proc. Int. Conf. Learn. Representations*, 2016, pp. 1–12.

[52] S. Lin *et al.*, "Accelerating convolutional networks via global & dynamic filter pruning," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 2425–2432.

Zhenzhen Wang received the B.E. and M.S. degrees from the School of Information and Control, Nanjing University of Information Science and Technology, Nanjing, China, in 2012 and 2015, respectively. She is currently working toward the Ph.D. degree with the School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore. Her current research interests include common pattern discovery, image retrieval, and network compression.



Weixiang Hong received the bachelor's degree in software engineering from Fudan University, Shanghai, China, in 2015, and the master's degree in software engineering from the National University of Singapore, Singapore, in 2019. Since 2015, he has been with Nanyang Technological University, Singapore under the supervision of Prof. J. Yuan. His current research interests include computer vision, and machine learning and optimization.

Yap-Peng Tan received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1993, and the M.A. and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ, USA, in 1995 and 1997, respectively. From 1997 to 1999, he was with Intel Corporation, Chandler, AZ, USA, and the Sharp Laboratories of America, Camas, WA, USA. In 1999, he joined Nanyang Technological University, Singapore, where he is currently a Professor and Acting Chair of the School of Electrical and Electronic Engineering (EEE). He was the Head of the Division of Information Engineering from 2005 to 2011 and an Associate Chair (Academic) of the School of EEE from 2011 to 2018, and the Founding Director of INFINITUS Infocomm Research Centre from 2011 to 2013. His current research interests include image and video processing, content-based multimedia analysis, computer vision, pattern recognition, machine learning, and data analytics. He is a member of the NTU Senate Committee on curriculum matters from 2013 to 2014. He served as the Chair for the Visual Signal Processing and Communications Technical Committee of the IEEE Circuits and Systems Society from 2012 to 2014, the Chair for the Membership and Election Subcommittee of the Multimedia Systems and Applications Technical Committee of the IEEE Circuits and Systems Society from 2013 to 2017, and the Chair for the Nominations and Elections Subcommittee of the Multimedia Signal Processing Technical Committee of the IEEE Signal Processing Society from 2012 to 2013. He was the Technical Program Co-Chair of ICME2018 and ICIP2019 and the Chair of the ICME Steering Committee from 2018 to 2019.

Junsong Yuan received the B.Eng. degree from the Special Program for the Gifted Young of the Huazhong University of Science and Technology (HUST), Wuhan, China, the M.Eng. degree from the National University of Singapore, Singapore, and the Ph.D. degree from Northwestern University, Evanston, IL, USA. He is currently an Associate Professor and the Director of the Visual Computing Lab, Department of Computer Science and Engineering (CSE), State University of New York at Buffalo, Buffalo, NY, USA. Before that he was an Associate Professor with Nanyang Technological University (NTU), Singapore. His research interests include computer vision, pattern recognition, video analytics, gesture and action analysis, large-scale visual search and mining. He received the Best Paper Award from the IEEE TRANSACTIONS ON MULTIMEDIA, Nanyang Assistant Professorship from NTU, and the Outstanding EECS Ph.D. Thesis Award from Northwestern University. He is currently a Senior Area Editor of the *Journal of Visual Communications and Image Representation* (JVCI), Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING (T-IP) and IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY (T-CSVT), and served as a Guest Editor of the *International Journal of Computer Vision* (IJCV). He is the Program Co-Chair of IEEE Conf. on Multimedia Expo (ICME'18) and Steering Committee Member of ICME (2018–2019). He also served as an Area Chair for CVPR and ACM MM'18, etc. He is a Fellow of the International Association of Pattern Recognition (IAPR).