Response to all reviewers: We thank all reviewers for their valuable and thoughtful comments. R #1 and R #4 describe our work as novel and interesting while requiring further concept explanation and performance comparison. R #2 appreciates the writing part but has concerns about the novelty. Below, we first clarify the novelty of this paper in response to the potential misunderstanding, then we present detailed answers for each reviewer.

Our main contributions are twofold: 1) We propose a new pruning paradigm called pruning filter in filter (PFF), which is a stripe-wise pruning and can be seen as a general case of the filter-pruning (FP). PFF treats a filter $F \in \mathbb{R}^{C\times K\times K}$ as $K \times K$ stripes, *i.e.*, 1×1 filters $\in \mathbb{R}^{C}$ and perform pruning at the unit of stripe instead of the whole filter. Compared to the existing methods, *i.e*, filter pruning (FP) and weight pruning (WP), PFF achieves finer granularity than traditional FP while being hardware friendly than WP, leading to state-of-the-art pruning ratio on CIFAR-10 and ImageNet. 2) More arousingly, by applying PFF, we find another important property of filters alongside their weight: *shape*. Start with a random initialized ResNet56, we train and trim the shape of filters and boost the test accuracy from 10.00% to 80.58% on CIFAR-10 without updating the filter weights. The optimal shape of filters are learned by the proposed Filter Skeleton (FS) in the paper and we believe FS could inspire further research towards the essence of network pruning.

Response to Reviewer #1: Q1: The paper does not compare to [1]. **A1:** [1] only uses MobileNet. However, we follow most pruning papers and use ResNet and VGG to perform experiments in the paper. Due to limited time during rebuttal, we only perform pruning with MobileNetV2 on ImageNet under PFF setting. PFF achieves 70.2% accuracy with 160M Flops, while [1] reported 67.1% accuracy with 167M Flops. We will conduct more experiments in the revised version. **Q2:** Inference time is not provided. **A2:** Since we modify the calculation order in convolution, only Python-version code (in the attached supplementary material) without re-implementing CUDA code can not reflect the actual inference time for PFF. In the future, we will re-implement CUDA code to provide the actual inference time of PFF. **Q3:** The difference between filter skeleton and [2]. **A3:** 1) [2] uses the binary mask to convert the full-stack filter to multiple sub-filters and one filter corresponds to multiple masks. However, FS learn the optimal shape of each filter and one filter corresponds to one mask; 2) The value of maks in [2] takes value from {-1,1}, thus all the weights of the filter are used. While the values of FS are continuous and are made sparse during training. The stripes corresponding to 0 in FS will be pruned; 3) The assumption in [2] is that the filters should be mutually orthogonal. While the assumption in FS is that the filters should be independent with each other. We will cite [1,2] and add these analyses to the paper properly.

Response to Reviewer #2: Q1: Is PFF the first to prune channel axes of convolution filter ?**A1:**[Kang, 2019, Accelerator-aware pruning for convolutional neural networks] does mention the concept of pruning channel axes (stripe) of convolution filters. However, they divide each stripe into fetching groups and perform weight pruning within each group, which is a hardware implementation optimization for weight pruning. To our best knowledge, PFF is the first stripe-wise pruning method. PFF only prunes the weight along channel axes (filter strip) and achieves state-of-the-art pruning ratio while being hardware friendly. We will cite this work and add these analyses to the paper properly. **Q2:** The experiments between shape-wise pruning and PFF are insufficient. **A2:** Besides the experiments in Table 1, we also perform experiments on shape-wise pruning and PFF in the supplementary material. In Figure 2 of the supplementary material, stripe-wise pruning (PFF) and shape-wise pruning are conducted in exactly the same setting. We can see that under the same number of parameters or Flops, PFF achieves a higher performance compared to shape-wise pruning. **Q3:** The difference between FS and soft mask in [24]. **A3:** Filter Skeleton (FS) in PFF is not just a soft mask. The mask in [24] is to learn the importance of each channel. While we consider each filter has two properties: weight and shape. FS is to learn the 'shape' property. From Section 4.3 in the paper, the network still has good performance by only learning the 'shape' of the filters, keeping the filter weight randomly initialized.

Response to Reviewer #4: Q1: what does the shape and optimal shape mean? **A1:** For a filter $F \in \mathbb{R}^{C\times K\times K}$. There are $K \times K$ stripes. Each stripe can be kept or removed. Thus there are 2^{K^2} shapes of this filter. The optimal shape of the filter is the filter with minimal stripes that keeps maximal useful information. **Q2:** Why the optimal shape matters? **A2:** In L112 $\rightarrow$ L119, network with an optimal shape and 12.64% random initialized weight still achieves high performance, which indicates that a good shape of filters can still learn useful knowledge from data. Thus the optimal shape matters in pruning. **Q3:** Why use dot product in Eq (1) in the paper? **A3:** Filters used for extracting features are produced by multiplying W and I (dot product) in Eq (1). Similar notation is used in *https://arxiv.org/abs/1908.02023*. **Q4:** Why PFF does not need fine-tuning? **A4:** Weights of each stripe and FS are multiplied (dot product) and jointly optimized. Thus the stripes corresponding to 0 value in FS have no contribution to the output, which can be removed accordingly. We find that the fine-tuned network and the network without fine-tuning end up with very similar results on CIFAR-10, indicating the effectiveness of the learned filter shape. Thus as mentioned in L177, a post fine-tuning is not necessary on CIFAR-10. **Q5:** what is the x-axes in Figure 5? Why the weights trained by PFF is more stable? **A5:** The x-axes is the number of filters. It can be seen that the weights trained by PFF are sparse and smooth, which have a low variance to input images, leading to stable outputs. **Q6:** what does one-shot fine-tuning means? **A6:** One-shot means that we only train the pruned network once until convergence. **Q7:** How to define top-1 in Figure 6? **A7:** Top-1 indicates the highest shape frequency in such layer. We will make all the presentations more clear in the revised version.