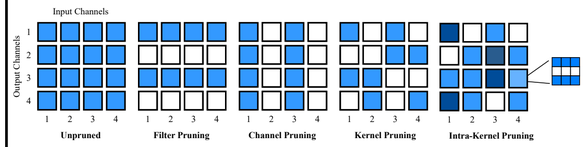
LeanFlex-GKP: Advancing Hassle-Free Structured Pruning with Simple Flexible Group Count

Jiamu Zhang[1], Shaochen (Henry) Zhong*[2], Andrew Ye*[1], Zirui Liu[2], Kaixiong Zhou[3], Xia Hu[2], Shuai Xu[1], Vipin Chaudhary[1]
[1]Case Western Reserve University [2]Rice University [3]Massachusetts Institute of Technology

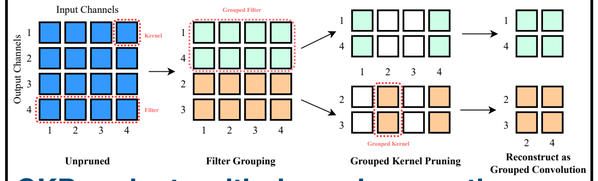
TL;DR: Pruning grouped kernels while remaining structured is great, but all performant grouped kernel pruning methods rely on dynamic operations with severe costs. We argue it is best to include such dynamic operations at `Conv2d(groups)`, resulting in a method with improved performance, efficiency, and user-friendliness.

Background

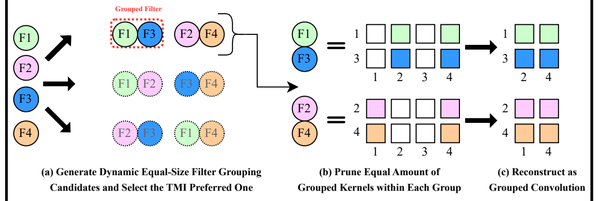
- Under the general realm of CNN network pruning, two categories of techniques have been proposed: **unstructured** pruning and **structured** pruning.
- **Unstructured** pruning:
  - Enjoy a higher **degree of pruning freedom** and thus better performance.
  - Result in a **sparse** network structure.
  - Require **special libraries or hardware** support to realize compression or acceleration benefits.
- **Structured** pruning:
  - Removes model components in **groups** that follow the architecture design.
  - Reduced in dimension yet **entirely dense**.
  - Provide **immediate** compression benefits without additional demand



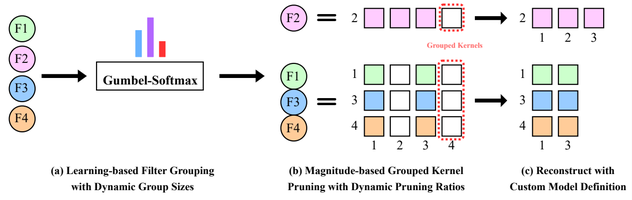
- To achieve better pruning freedom while remain densely structured, a special type of intra-channel pruning granularity called **Grouped Kernel Pruning (GKP)** has been proposed in ICLR 2022.



GKP-variants with dynamic operations



- **TMI-GKP** opts to include dynamic choices of *clustering schemes* in each of its convolutional layers.
- Some *clustering schemes* involving dimensionality reduction can be very expensive to run (e.g., k-PCA).
- Requires training snapshots or checkpoints of the unpruned model, which is not user friendly in practical applications.



- **DSP** makes its filter grouping and group kernel pruning stages **dynamic** in the sense that they may enjoy **different group sizes** and **different in-group pruning rates** within the same layer.
- But resultant pruned network is **irregularly shaped** and therefore relies on **custom** model definitions and convolutional operators for fine-tuning and inference.

What we have done?

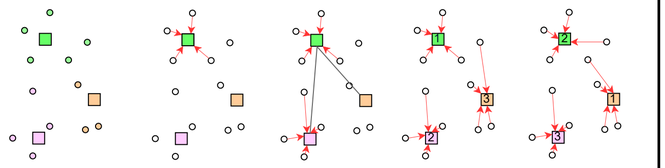
- We propose a new method to include the **dynamic operation** within `Conv2d(groups)` (a.k.a. "**group count**"), and argue that allowing each convolutional layer to take a **flexible number of groups** when grouping filters is the **best area to integrate dynamic operations** into a GKP procedure.
- Our proposed method, **LeanFlex-GKP**, consists of a **four-stage** procedure:
  - **Filter grouping:**
    - group filters within a certain layer into n equal-sized filter groups according to their distance towards k-Means++ determined centers.
  - **Group kernel pruning**
    - prune a certain amount of grouped kernels out of all filter groups within the same layer, determined by each grouped kernel's L2 norm and distance to their geometric median.
  - **Post-prune group count evaluation**
    - evaluate all grouping and pruning strategies obtained under different group count settings and select the one where the preserved group kernels have the maximum inter-group distance and the minimum intra-group distance.
  - **Grouped convolution reconstruction**
    - convert the pruned model to a grouped convolution format.

Contributions

- **Advance the progress of GKP** by identifying and solving a common pain points — dynamic operation.
- Provide an **efficient**, **hassle-free experience** by proposing a method that is post-train, one-shot, data-agnostic, and with just one tunable hyper-parameter.
  - One can also measure the compute/memory requirement of a pruned model before the actual pruning, as well as be able to prune with different aggressiveness — two **user-friendly characteristics** (surprisingly) lacking in many modern pruning methods.
- Massive **speed advantage** on pruning procedure over existing performant GKP variants.
- Achieve SOTA-competitive and even beyond **SOTA performance** on wide range of CNN model architecture and dataset.
- **Guiding future developments of GKP** with our design insights and ablation studies.

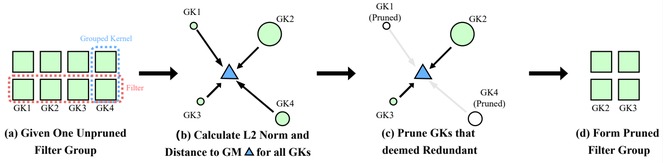
How we did it?

#1: KPP-Induced Filter Grouping



- We first cluster filters (the circles) via KPP into n groups with no constraint on having an equal group size to determine clustering centers (the squares), as in (a).
- Then, our operation can be viewed as a cycle between assigning m nearest filters into a KPP center to form a filter group, then finding the next KPP center to do subsequent filter assignments, as in (b) → (c); until n filter groups are formed (the first KPP center is picked at random).
- Last, we conduct a multiple restart and repeat (b) and (c) center-finding-filter-assignments, as showcased in (d).
- After all multiple restarts, we are left with n candidate filter grouping strategies, and select the strategy that has filters with the least intra-group distance to their respective KPP centers (having less summed length on red arrows).

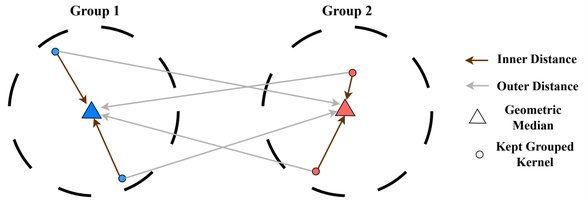
#2: GM + L2 Grouped Kernel Pruning



- Previous methods like TMI-GKP converted its grouped kernel selection problem as a graph search problem, added with the help of a greedy procedure and multiple restarts.
- While such a procedure is generally efficient, it is still time and resource-consuming given a wide layer.

- We utilize a simple combination of L2 norm and Geometric Median based distance to form a lightning-fast pruning procedure:
  - Given an unpruned filter group as in (a), we first calculate the Geometric Median (GM) of its Grouped Kernels (GKs), as well as each GK's distance to the GM and their L2 norm.
  - These distances and the L2 norm are visualized in (b) as the length of black arrows and the area of green circles, respectively.
  - The GKs with large L2 norms and small distances to their GMs are preserved and eventually reconstructed to the grouped convolution format, as shown (c) to (d).

#3: Post-Prune Group Count Evaluation



- We first compute the GM among retained grouped kernels and then calculate the inner and outer distance among them.
- After a normalization w.r.t. the group count, the one with the highest average (Outer Distance - Inner Distance) is chosen.
- Given each group count evaluation is conducted upon a pruned conv layer (after being grouped with different `Conv2d(groups)`), our method makes connections between the (originally independent) filter grouping and grouped kernel pruning stage.

Main Experiments (abbreviated)

Method	DA	IP	RB	BA	Pruned	ΔAcc	↓MACs	↓Params
ResNet110 on CIFAR10	MACs ≈ 255.0M			Params ≈ 1.73M				
TMI-GKP (Zhong et al., 2022)	✓	✗	300	94.26	94.90	↑0.64	43.31	43.52
L1Norm-B (Li et al., 2017)	✓	✗	300	94.26	92.96	↓1.30	43.17	36.69
CC (Li et al., 2021)	✗	✗	300	94.26	94.31	↑0.05	44.54	39.47
SFP (He et al., 2018c)	✗	✓	300	94.26	94.44	↑0.18	43.42	43.52
GAL (Lin et al., 2019a)	✗	✓	300	94.26	93.42	↓0.84	29.14	31.37
FPGM (He et al., 2019)	✗	✓	300	94.26	94.18	↓0.08	43.39	43.52
NPPM (Gao et al., 2021)	✗	✗	300	94.26	94.16	↓0.10	42.46	35.19
HRank (Lin et al., 2020)	✗	✓	300	94.26	92.96	↓1.30	18.57	5.38
DHP (Li et al., 2020)	✗	✓	300	94.26	92.53	↓1.73	60.25	64.58
LRF (Joo et al., 2021)	✗	✗	300	94.26	94.49	↑0.23	43.37	42.30
OTOv2 (Chen et al., 2023)	✗	✓	300	94.26	91.58	↓2.68	37.83	42.44
KPGP* (Zhang et al., 2022b)	✓	✗	300	93.76	94.01	↑0.25	43.3	43.5
LeanFlex-GKP (ours)	✓	✗	300	94.26	**94.92**	↑**0.66**	43.31	43.52
ResNet110 on CIFAR100	MACs ≈ 255.001M			Params ≈ 1.734M				
TMI-GKP (Zhong et al., 2022)	✓	✗	300	72.99	72.79	↓0.20	43.31	43.37
L1Norm-A (Li et al., 2017)	✓	✗	300	73.20	69.85	↓3.35	43.74	44.41
CC (Li et al., 2021)	✗	✗	300	73.20	73.21	↑0.01	43.43	19.78
NPPM (Gao et al., 2021)	✗	✗	300	73.20	72.38	↓0.82	42.77	18.69
LRF (Joo et al., 2021)	✗	✗	300	73.20	73.58	↑0.38	43.38	42.16
LCCL* (Dong et al., 2017)	✗	-	300	72.79	70.78	↓2.01	31.3	-
SFP* (He et al., 2018a)	✗	✓	300	74.14	71.28	↓2.86	52.3	-
FPGM* (He et al., 2019)	✗	✓	300	74.14	72.55	↓1.59	52.3	-
TAS* (Dong & Yang, 2019)	✗	✗	300	75.06	73.16	↓1.90	52.6	-
LeanFlex-GKP (ours)	✓	✗	300	73.20	**73.63**	↑**0.43**	43.31	43.36
ResNet56 on Tiny-ImageNet	MACs ≈ 506.254M			Params ≈ 0.865M				
TMI-GKP (Zhong et al., 2022)	✓	✗	300	56.13	55.52	↓0.61	37.05	36.76
L1Norm-A (Li et al., 2017)	✓	✗	300	56.13	55.41	↓0.72	35.51	32.14
SFP (He et al., 2018a)	✗	✓	300	56.13	53.65	↓2.48	33.96	35.38
FPGM (He et al., 2019)	✗	✓	300	56.13	54.14	↓1.99	33.53	34.68
HRank (Lin et al., 2020)	✗	✓	300	56.13	54.16	↓1.97	37.39	30.98
GAL* (Lin et al., 2019b)	✗	✓	100	56.55	55.87	↓0.68	52.00	32.00
DHP* (Li et al., 2020)	✗	✓	100	56.55	55.82	↓0.73	55.00	46.00
3D* (Wang et al., 2021)	✗	✓	420	56.55	**56.04**	↓0.51	59.00	34.00
Slimming* (Liu et al., 2017)	✗	✗	100	56.55	52.45	↓4.10	53.00	54.00
LeanFlex-GKP (ours)	✓	✗	300	56.13	**55.67**	↓**0.46**	37.05	36.76
ResNet50 on ImageNet-1K	MACs ≈ 4122.828M			Params ≈ 25.557M				
SFP* (He et al., 2018a)	✗	✓	100	76.13	58.50	↓17.63	36.08	32.31
FPGM* (He et al., 2019)	✗	✓	100	76.13	75.04	↓1.09	35.93	28.36
TMI-GKP* (Zhong et al., 2022)	✓	✗	100	76.15	75.53	↓0.62	33.21	33.74
ThiNet* (Luo et al., 2017)	✗	✓	100	72.88	72.04	↓0.84	36.7	-
OTOv2* (post-train) (Chen et al., 2023)	✗	✓	120	76.13	75.38	↓0.75	37.70	26.58
DOP* (Yang et al., 2022)	✗	✗	120	76.47	74.29	↓2.18	60.00	-
KPGP* (Zhang et al., 2022b)	✓	✗		76.15	75.50	↓0.65	33.7	33.2
Layer-wise Proxy* (Elkerdawy et al., 2020)	✗	✗	-	76.14	75.0	↓1.14	5.5	-
LeanFlex-GKP (ours)	✓	✗	100	76.13	**75.62**	↓**0.51**	33.06	30.34