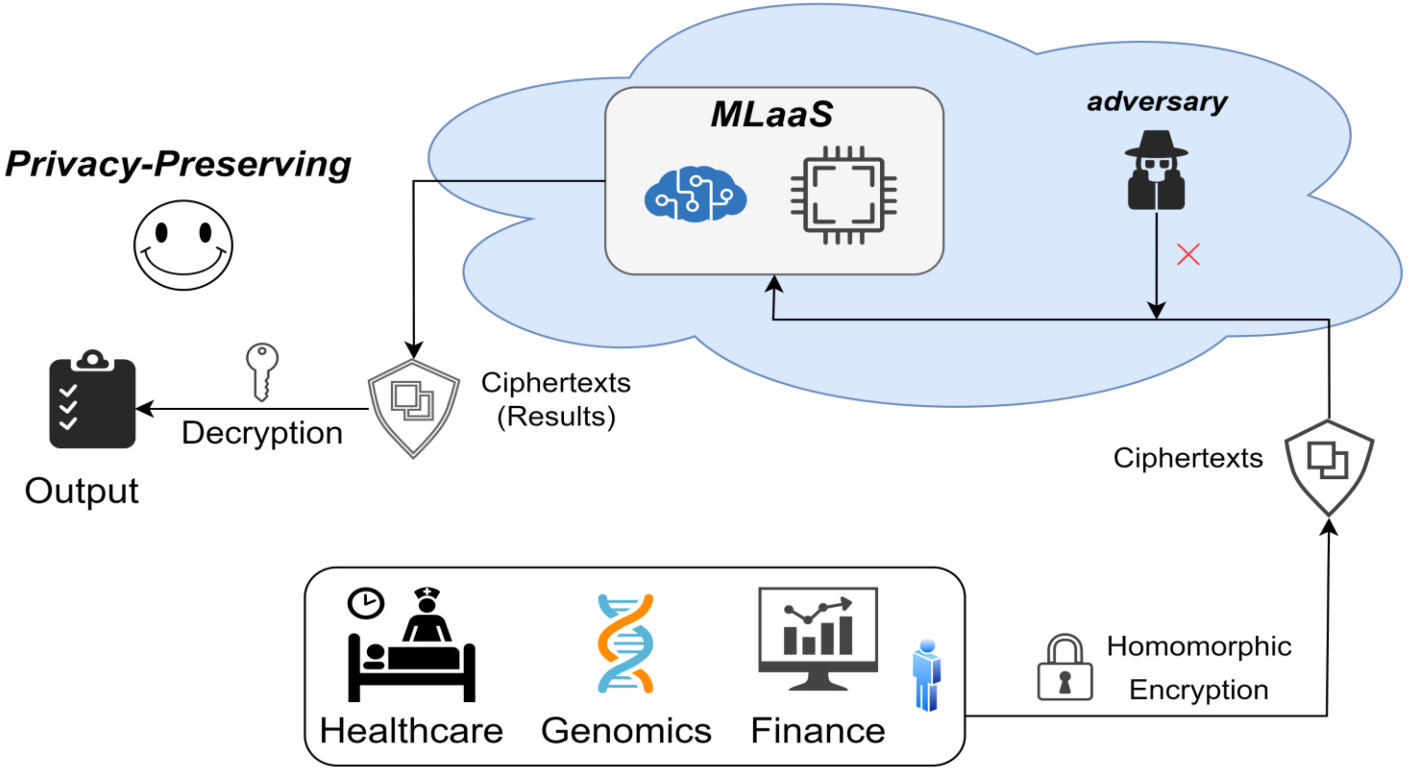
SpENCNN: Orchestrating Encoding and Sparsity for Fast Homomorphically Encrypted Neural Network Inference

Ran Ran[1], Xinwei Luo[1], Wei Wang[2], Tao Liu[3],
Gang Quan[4], Xiaolin Xu[5], Caiwen Ding[6], Wujie Wen[1]
Lehigh University[1], Anonym, Inc[2], Lawrence Technological University[3],
Florida International University[4], Northeastern University[5] ,
University of Connecticut[6]

To appear @ ICML 2023
Source code: https://github.com/ranran0523/SpENCNN

Homomorphic Encryption - PPML



CKKS scheme (for real number)

Encoding Process

- Message vector
- Notated as: $(Z_0,\ldots,Z_{N/2-1})$

→

- Packed in polynomial
- Notated as: $p(x)$

Encrypt Process

- Encrypt to Ciphertext $c(x)$
- $c(x)=(c_0(x),c_1(x)) = v*(b,a)+(m+e_0, e_1)$

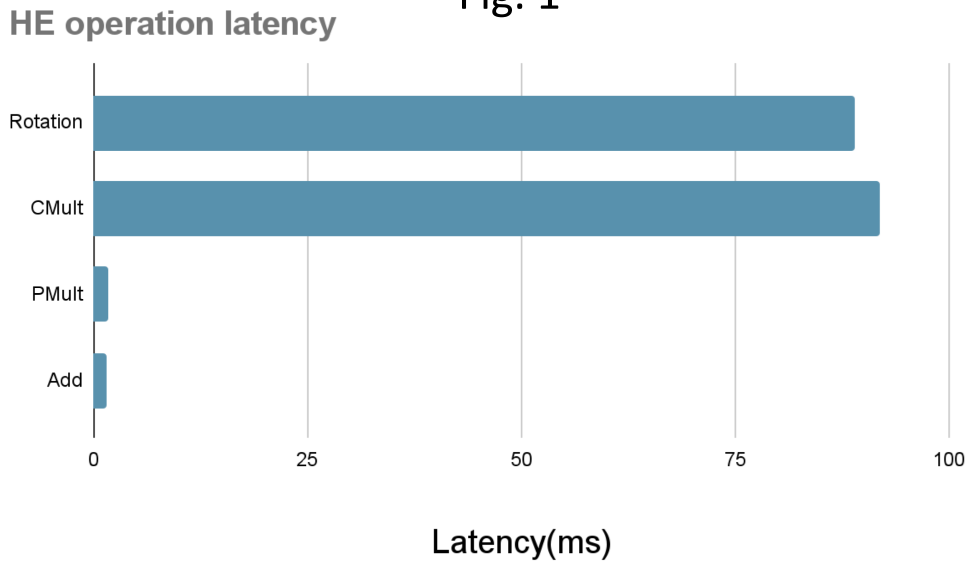
→

- Decrypt to plaintext $m(x)$
- $m(x)=c_0(x)+c_1(x)*s$

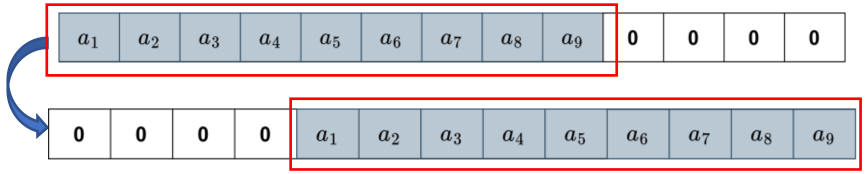
Where v, e_0, e_1 are random polynomials

Our Observations - Bottlenecks

Fig. 1



Main problem = Less Rotation !!!



e.g. Rotation for 4 slots

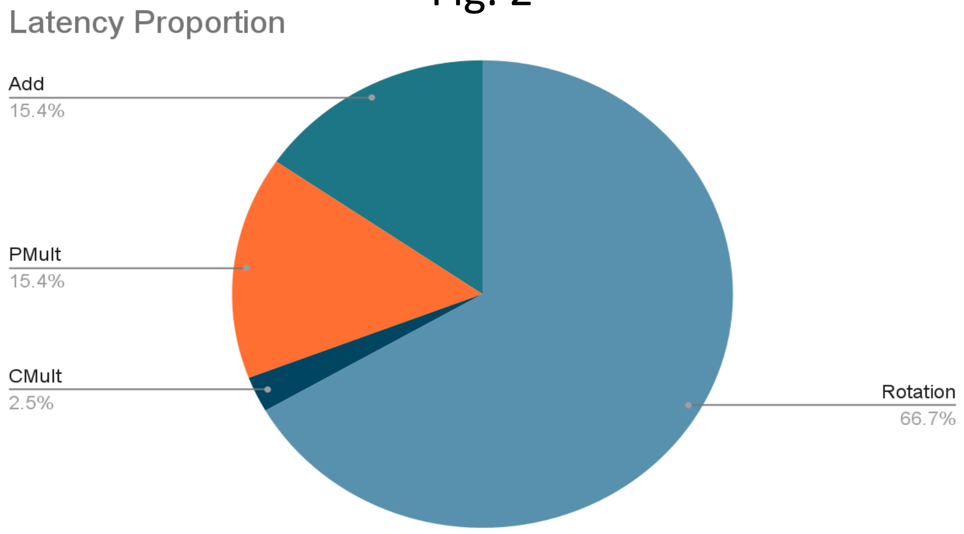
Rotation and **CMult** contain **Key-Switching (KS)** operation which lead to a high latency than others [1].

Supported HE operations in CKKS:

Rot(c(x),k)= (1,2,3,…,n) -> (k,k+1,…n,1,2,…,k-1)

CMult(c(x), c'(x))=c(x) * c'(x)
PMult(c(x), p(x))= c(x) * p(x)
Add(c(x), c(x))= c(x) + c'(x)

Fig. 2



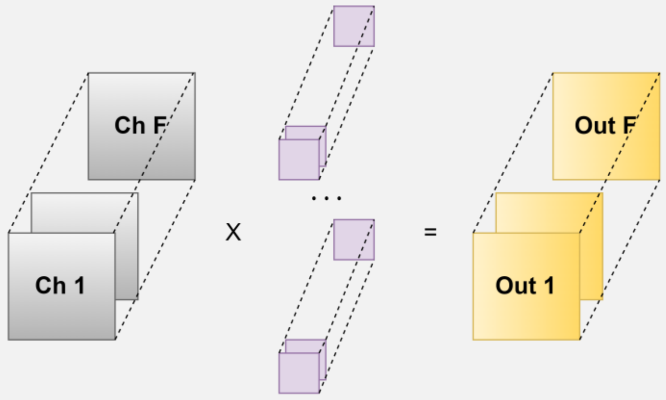
One 64-channel Convolutional Layer Profiling Result

[1] Jiang, Xiaoqian, et al. "Secure outsourced matrix computation and application to neural networks." Proceedings of the 2018 ACM SIGSAC conference on computer and communications security. 2018.

CNN Computation Pattern in HE

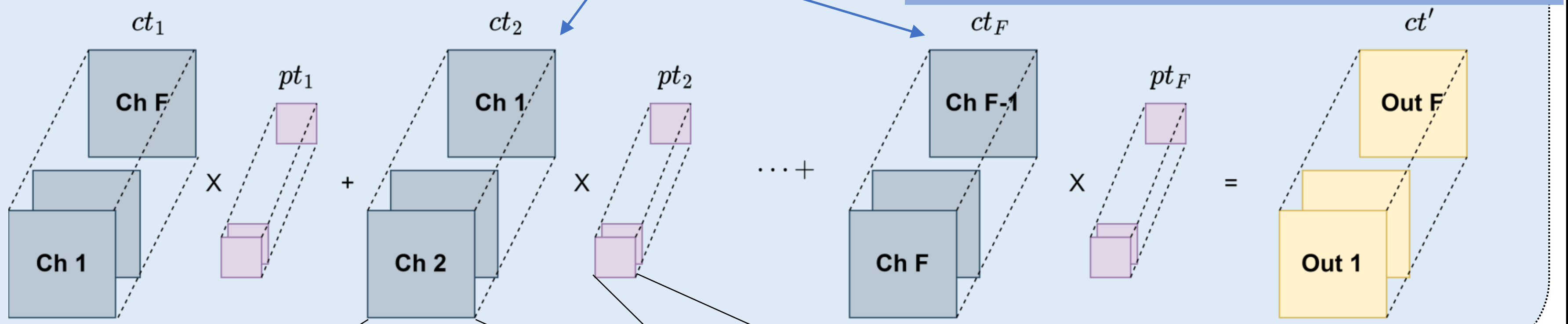
LEHIGH UNIVERSITY

Plaintext Convolution



Outer-rotation generated copies

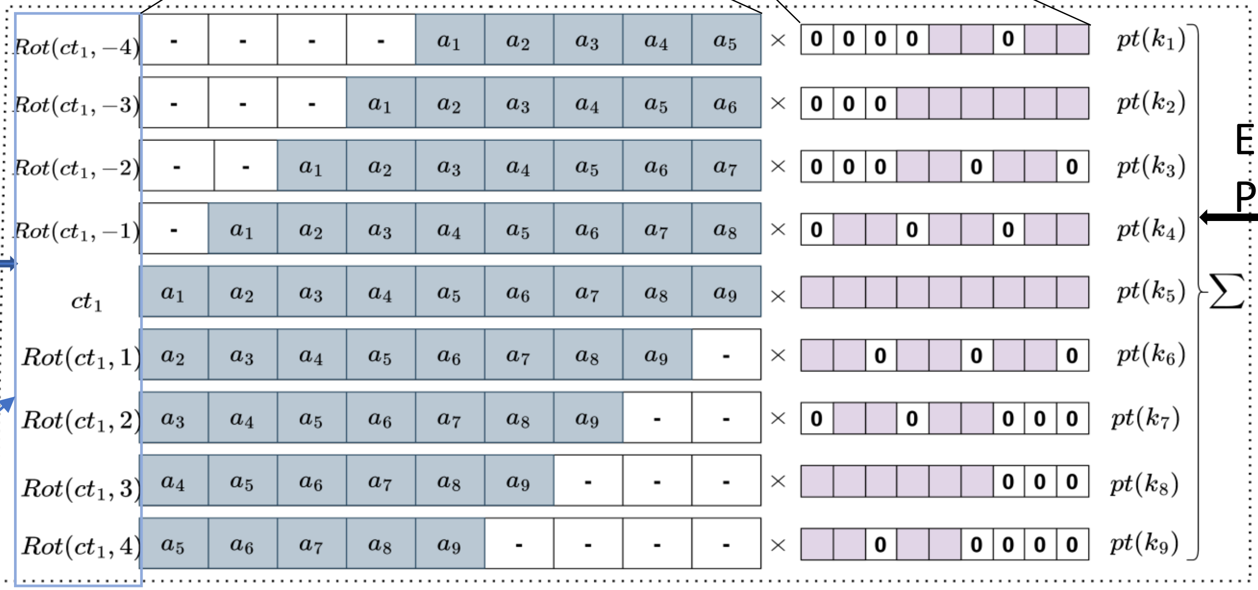
# of Outer-rotations = ***F-1***
(F=# of channels on ciphertext)



General HE-Convolution with 3x3 kernel
(**One Channel Example**)

a_1	a_2	a_3
a_4	a_5	a_6
a_7	a_8	a_9

Encrypt and Rotate



Encode to Plaintext

k_1	k_2	k_3
k_4	k_5	k_6
k_7	k_8	k_9

# of Inner-rotations = ***K^2-1***
(K=Kernel Size)

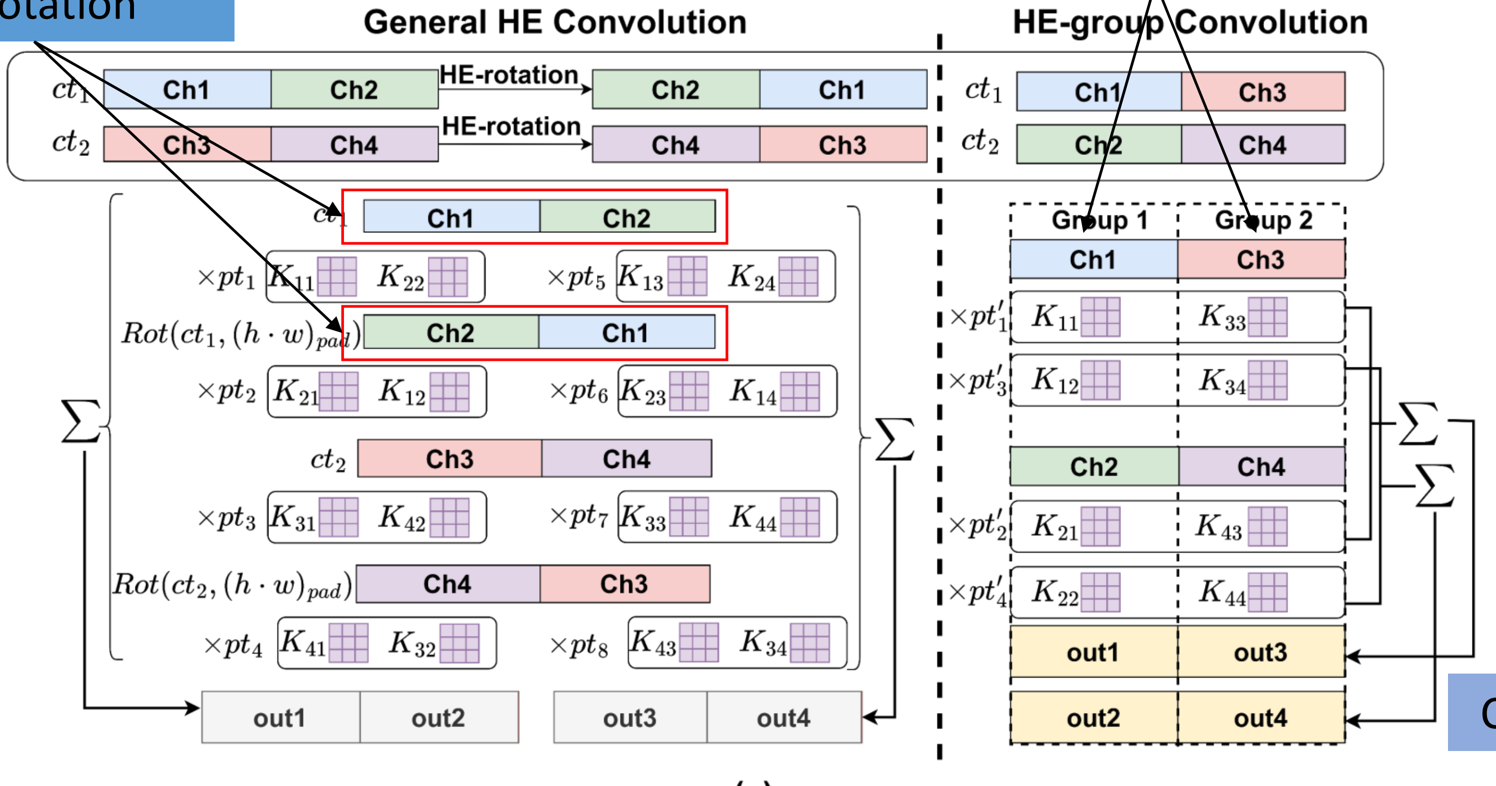
HE-Group Convolution

How to skip Intensive HE operations, e.g. Rotation?

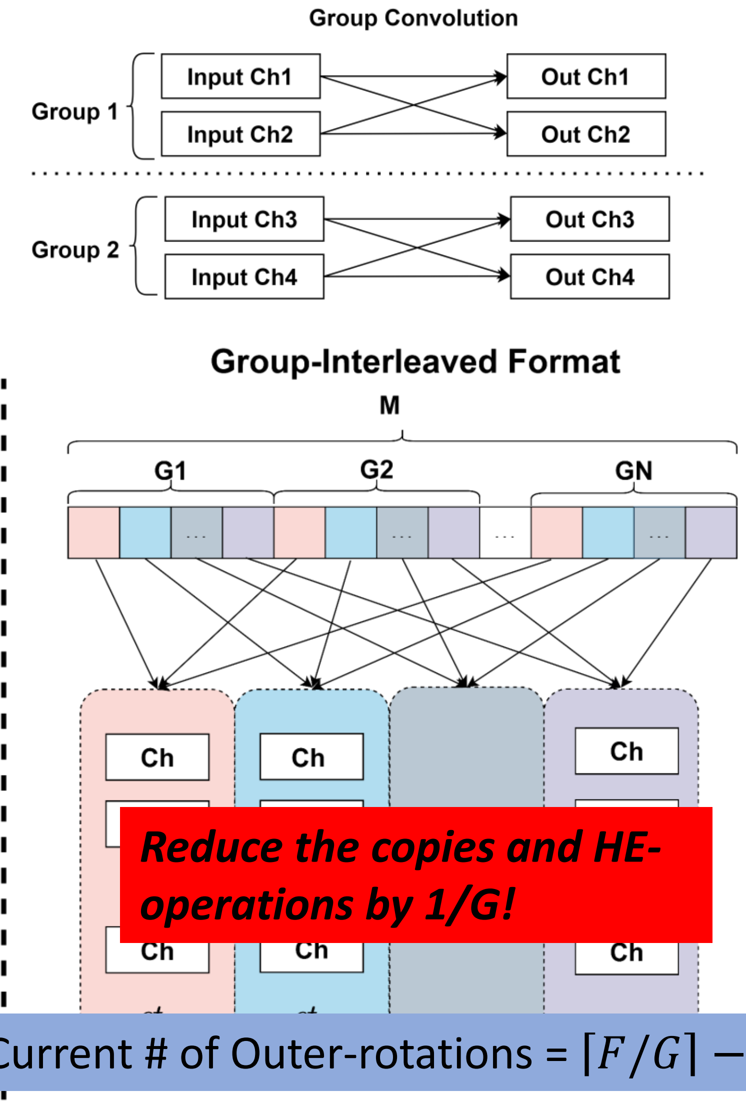
Rethink the CNN computation pattern in HE domain

Less Ciphertext Copies Needed (**Independent Channels**)

Outer-rotation



(a)



(b)

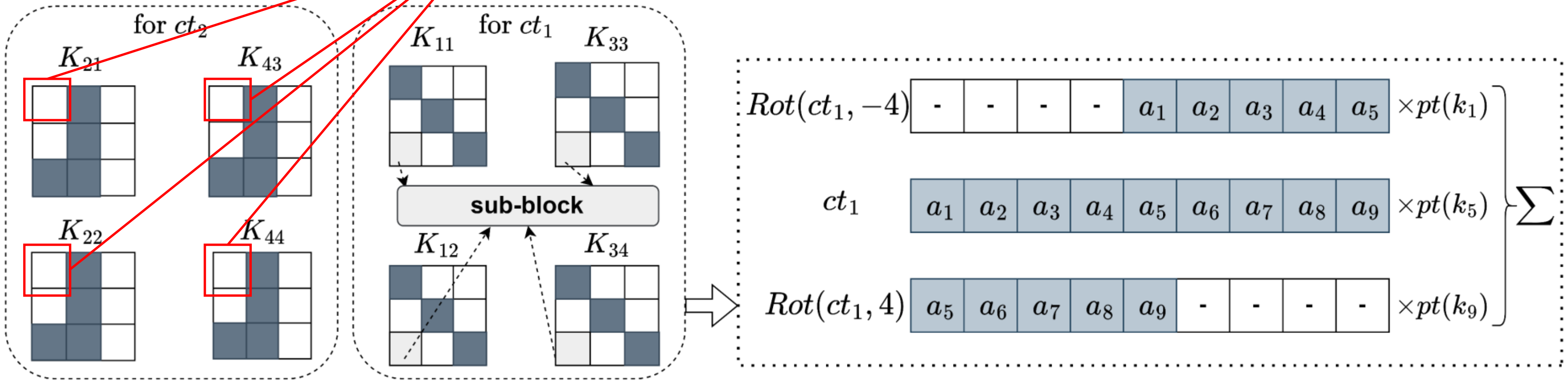
Reduce the copies and HE-operations by 1/G!

Current # of Outer-rotations = $\lceil F/G \rceil - 1$

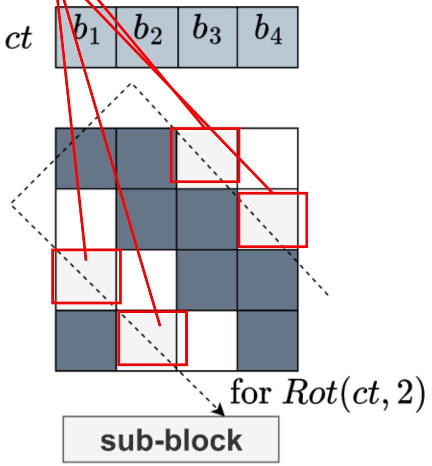
Sub-Block Pruning

HE-Block Configuration (kernel weights in same location)

HE-Block Configuration (weights in diagonal wise)

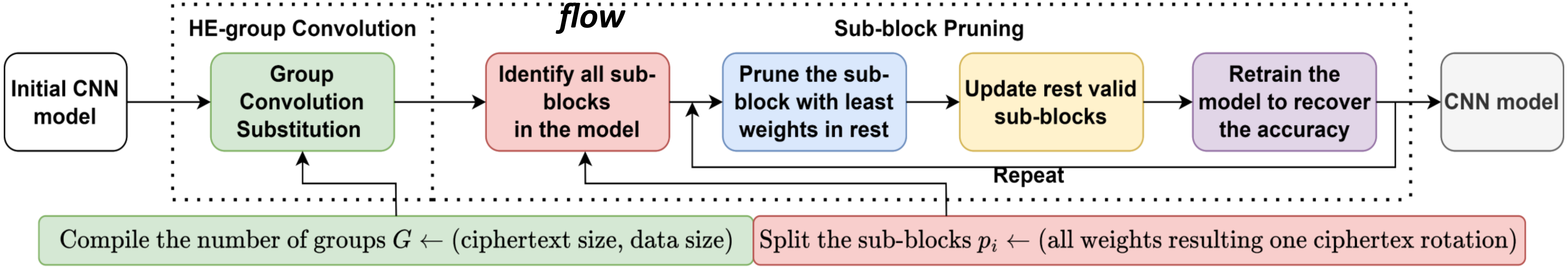


(a) Weight sparsity in convolutional layers



(b) Weight sparsity in FC layers

Overview of combined optimization flow



Experiment Results

LEHIGH UNIVERSITY

Table 3. Ablation study of HE-group convolution with the different number of convolution groups.

Model	Groups	HOC Left (%) Rot	Others	Accuracy (%)	Latency (s)	Speedup (×)
LeNet-like	1-baseline	-	-	98.95	1.2658	-
	2	51.52	52.91	98.95	0.6806	1.86
	4	**27.27**	**28.24**	**98.95**	**0.3807**	**3.32**
	8	27.27	16.47	98.67	0.3044	4.16
VGG-5	1-baseline	-	-	85.16	53.909	-
	4	87.53	84.08	84.53	46.539	1.16
	8	**85.45**	**81.42**	**84.06**	**45.311**	**1.19**
	16	85.45	80.10	82.23	45.053	1.20
HEFNet	1-baseline	-	-	84.91	24.113	-
	4	24.53	25.74	84.35	6.2491	3.86
	8	**11.95**	**13.36**	**83.67**	**3.2718**	**7.37**
	16	11.95	7.18	80.06	2.3627	10.21
ResNet-20	1-baseline	-	-	91.52	647	-
	2	51.4	52.72	91.43	475	1.36
	4	**27.11**	**28.76**	**90.21**	**392**	**1.65**
	8	14.96	15.12	85.31	351	1.84

Table 4. Ablation study of sub-block prune and comparison with other pruning methods.

Network	Groups	HOC Left (%) Rot	Others	Sparsity (%)	Latency (s)	Speedup (×)
LeNet-like	Dense-Baseline	-	-	0.00	1.2658	-
	NS-prune	96.12	96.23	91.00	1.2190	1.04
	S-prune (channel)	88.03	92.82	53.77	1.1202	1.13
	Sub-block prune	**35.21**	**34.07**	**63.83**	**0.4644**	**2.62**
VGG-5	Dense-Baseline	-	-	0.00	53.909	-
	NS-prune	97.59	97.14	91.88	52.5280	1.03
	S-prune (channel)	98.47	98.08	90.48	50.7178	1.06
	Sub-block prune	**15.89**	**16.11**	**89.87**	**8.7659**	**6.15**
HEFNet	Dense-Baseline	-	-	0.00	24.113	-
	NS-prune	85.60	88.97	72.95	21.1660	1.14
	S-prune (channel)	94.69	95.24	51.91	22.9240	1.05
	Sub-block prune	**41.88**	**38.11**	**63.90**	**9.3709**	**2.57**
ResNet-20	Dense-baseline	-	-	91.52	647	-
	NS-prune	90.23	91.82	78.21	599	1.08
	S-prune (channel)	96.21	96.84	53.12	628	1.03
	Sub-block prune	**52.31**	**50.12**	**56.40**	**475**	**1.36**

Table 5. Comparison with Hunter on model HOC left, sparsity, accuracy, latency, and speedup.

Network	Method	HOC Left (%) Rot	Others	Sparsity (%)	Accuracy (%)	Latency (s)	Speedup (×)
LeNet-like	Baseline	-	-	0	98.95	1.2658	-
	Hunter	40.95	39.91	59.99	98.95	0.5353	2.36
	Ours-4	**8.54**	**9.88**	**62.62**	**98.95**	**0.1535**	**8.37**
VGG-5	Baseline	-	-	0	85.16	53.909	-
	Hunter	17.86	18.93	89.81	84.03	9.9916	5.40
	Ours-8	**7.86**	**7.72**	**91.97**	**84.07**	**4.3830**	**12.11**
HEFNet	Baseline	-	-	0	84.91	24.113	-
	Hunter	48.27	42.20	57.82	83.63	10.855	2.22
	Ours-8	**3.99**	**4.61**	**65.62**	**83.67**	**1.2520**	**19.26**
ResNet-20	Baseline	-	-	0	91.52	647	-
	Hunter	51.12	52.39	48.12	90.20	461	1.40
	Ours-4	**14.10**	**15.47**	**53.32**	**90.21**	**344**	**1.87**

still Effective for
with bootstrapping

Conclusion and Future Work

1.To conclude our work, we first combine the HE encoding format and the group convolution to reduce inference latency.

2. We rethink the sparsity problem in HE domain and structurally prunes weights by one sub-block for one high-latency inner-rotation operation

3. Future work could be extended to other applications and combines with other optimization methods like quantization to achieve a further reduction of latency.

Thanks!

Welcome to my poster for more discussions.