EfficientViT: Memory Efficient Vision Transformer with Cascaded Group Attention

——— Supplementary Material ———

Xinyu Liu[1,*] , Houwen Peng[2], Ningxin Zheng[2], Yuqing Yang[2], Han Hu[2], Yixuan Yuan[1]
[1] The Chinese University of Hong Kong, [2] Microsoft Research
1155195604@link.cuhk.edu.hk, {houwen.peng, ningxin.zheng, yuqing.yang, hanhu}@microsoft.com, yxyuan@ee.cuhk.edu.hk

This supplementary material presents additional details of Section 2.1, 2.3, 4.2, and 4.3. Besides, extra experiments show that EfficientViT can be further accelerated using automatically searched kernel with TVM [3].

- **Runtime Profiling on Subnetworks.** We present the runtime profiling of subnetworks in Sec. 2.1.
- **Parameter Efficiency Analysis for DeiT-T.** We provide the results of using Taylor pruning [10] on DeiT-T [14] for parameter efficiency analysis in Sec. 2.3.
- **Comparison on Mobile Chipsets.** We deploy our model on Apple A13 Bionic chip in iPhone 11 with CoreML [1], and compare with other efficient models designed for mobiles in Sec. 4.1.
- **Instance Segmentation.** We present results on COCO instance segmentation benchmark [7] and compare with other efficient models in Sec. 4.2.
- **Further Acceleration with TVM.** We apply automatic kernel optimization with TVM [3] and show that EfficientViT can be further accelerated.

A. Runtime Profiling on Subnetworks

We perform runtime profiling for subnetworks in Sec. 2.1 in the main manuscript, and present the results of Swin-T-1.25×, Swin-T-1.5×, DeiT-T-1.25×, and DeiT-T-1.5× in Fig. 1, 2, 3, and 4, respectively. It is observed that under a similar inference throughput, the subnetworks with smaller proportions of MHSA layers tend to have less time consumption on memory-bound operations. The results further validate that reducing the utilization ratio of MHSA layers appropriately can enhance memory efficiency.

B. Parameter Efficiency Analysis for DeiT-T

To further study the parameter redundancy in vision transformers in Sec. 2.3, we also adopt Taylor structured pruning [10, 15] to automatically find the important modules in DeiT-T [14]. The ratios between the remaining output channels to the input embedding dimensions are plotted in Fig. 5, and the original ratios in the unpruned model are also given for reference. Similar to the pruning results of Swin-T in Sec. 2.3, we observe that the Q,K dimensions are largely trimmed, whereas V prefers relatively large channels, being close to the input embedding dimension. The only difference is that the FFNs in DeiT-T are less likely to get pruned, which demonstrates that the channel redundancy in the isomorphic structure may be less significant than in the hierarchical structure. Meanwhile, it is shown that the model tends to preserve more channels in FFN than MHSA, which suggests the importance of channel communication in vision transformers, and may further reflect the effectiveness of the proposed sandwich layout design.

C. Comparison on Mobile Chipsets

To test the performance on mobile devices, we deploy the proposed EfficientViT on the mobile chipset, *i.e.*, Apple A13 Bionic chip in iPhone 11, and provide the results in Tab. 1. We compare our EfficientViT with other efficient models that were designed for mobiles, including MobileViT [9] and MobileNetV3 [6]. CoreMLTools [1] is used for the deployment. Compared to MobileViT-XXS, EfficientViT-M2 runs 2.3× faster with 1.8% higher accuracy. Compared to the state-of-the-art efficient CNN MobileNetV3, EfficientViT-M4 has comparable accuracy yet runs 9.7% faster, and achieves 1.9% higher accuracy when trained for 1,000 epochs with distillation as in Sec. 4.4. The results demonstrate the proposed design is efficient across different deployment platforms.

D. Instance Segmentation

We use Mask R-CNN [5] with FPN for instance segmentation task on COCO [7], and train the models for 12 epochs (1× schedule) with the same settings as [8] on MMdetec-

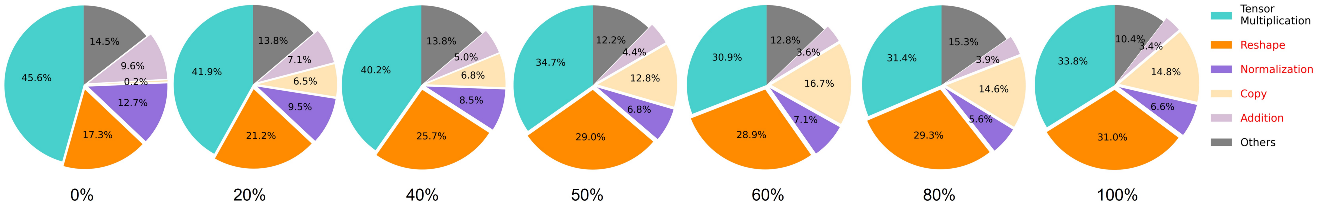

Figure 1. Runtime profiling of different subnetworks of Swin-T with 1.25× acceleration. Red text denotes memory-bound operations. The percentages below the figures denote the MHSA layer proportions.

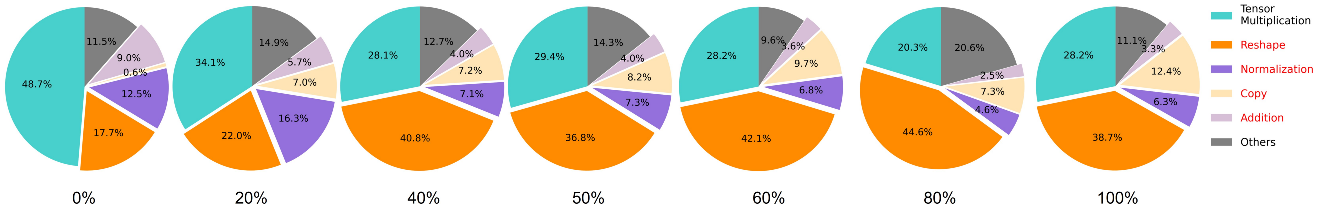

Figure 2. Runtime profiling of different subnetworks of Swin-T with 1.50× acceleration. Red text denotes memory-bound operations. The percentages below the figures denote the MHSA layer proportions.

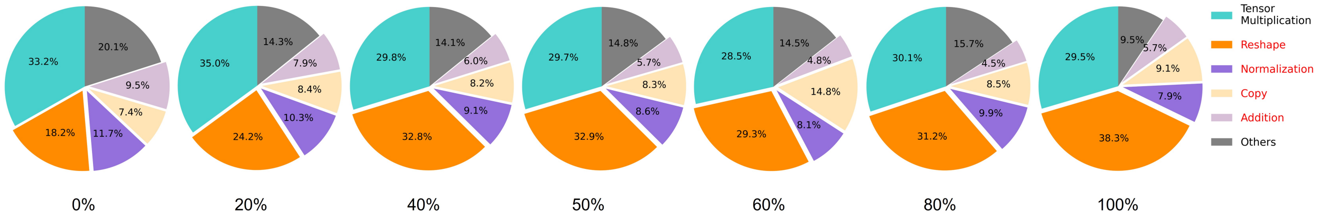

Figure 3. Runtime profiling of different subnetworks of DeiT-T with 1.25× acceleration. Red text denotes memory-bound operations. The percentages below the figures denote the MHSA layer proportions.

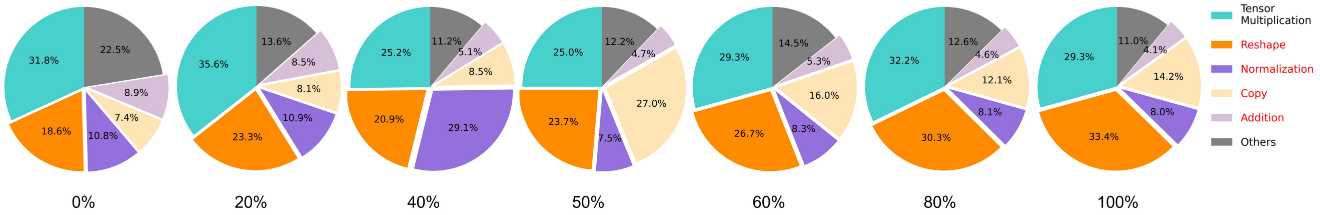

Figure 4. Runtime profiling of different subnetworks of DeiT-T with 1.50× acceleration. Red text denotes memory-bound operations. The percentages below the figures denote the MHSA layer proportions.

tion [2]. Specifically, to adapt the 3 backbone features with strides 16, 32, and 64 in EfficientViT to FPN, we apply 2 deconvolutions on the stride 16 feature to generate 2 additional features with strides 8 and 4. Then, these 5 features are fed to the FPN. We compare EfficientViT-M4 with other efficient models and present the results in Tab. 2. Compared to MobileNetV2, our EfficientViT-M4 uses comparable Flops yet achieves 3.2% higher AP^b and 3.8% higher AP^m, respectively. Compared to the prevailing searched efficient model EfficientNet-B0, our model surpasses it by 0.9% in AP^b and 1.6% in AP^m, while using 42.47% fewer Flops, demonstrating the transfer ability of the proposed model.

E. Further Acceleration with TVM

To further accelerate the proposed EfficientViT on CPU, we propose to apply automatic kernel optimization with TVM [3], and show the results in Tab. 3. With the automatically searched kernel, the proposed models further show remarkable throughput improvements. *e.g.*, the throughput of EfficientViT-M1 is increased by 71.4% on an Intel Xeon

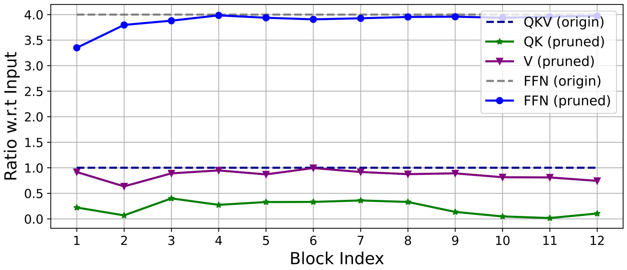

Figure 5. The ratio of the channels to the input embeddings before and after pruning DeiT-T [14]. Baseline accuracy: 67.0%; pruned accuracy: 59.6%.

Table 1. CoreML [1] performance of EfficientViT and other efficient models designed for mobiles. The result in brackets is trained for 1,000 epochs with distillation.

Model	Top-1 (%)	Latency (ms)	Flops (M)	Params (M)
MobileViT-XXS [9]	69.0	12.03	410	1.3
MobileViT-XS [9]	74.7	23.01	986	2.3
MobileNetV3 [6]	75.2	7.43	217	5.4
EfficientViT-M2	70.8	5.23	201	4.2
EfficientViT-M4	74.3 (77.1)	6.71	299	8.8
EfficientViT-M5	77.1	8.64	522	12.4

Table 2. EfficientViT instance segmentation performance on COCO `val2017` [7] with comparisons to other efficient models.

Model	Mask R-CNN 1×						Flops	Params
	AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}	(M)	(M)
MobileNetV2 [11]	29.6	48.3	31.5	27.2	45.2	28.6	300	3.4
MobileNetV3 [6]	29.2	48.6	30.3	27.1	45.5	28.2	217	2.8
FairNas-C [4]	31.8	51.2	33.8	29.4	48.3	31.0	325	5.6
EfficientNet-B0 [13]	31.9	51.0	34.5	29.4	47.9	31.2	522	3.6
MNASNet-A1 [12]	32.1	51.9	34.2	29.7	49.0	31.4	312	3.9
EfficientViT-M4	**32.8**	**54.4**	**34.5**	**31.0**	**51.2**	**32.2**	299	8.8

Table 3. CPU throughput of EfficientViT family without and with TVM [3] kernel optimization.

Model	Top-1	Throughput (imgs/s)	
	(%)	CPU	CPU (TVM)
EfficientViT-M0	63.2	228.4	366.8 (+60.6%)
EfficientViT-M1	68.4	126.9	217.5 (+71.4%)
EfficientViT-M2	70.8	121.2	182.0 (+50.2%)
EfficientViT-M3	73.4	96.4	142.2 (+47.5%)
EfficientViT-M4	74.3	88.5	126.0 (+42.4%)
EfficientViT-M5	77.1	56.8	78.5 (+38.2%)

E5-2690 v4 @ 2.60 GHz processor. The results demonstrate the potential of EfficientViT in achieving much faster inference speed with the optimization of the kernel functions.

References

[1] Apple. Coremltools: Use coremltools to convert machine learning models from third-party libraries to the core ml format, 2021. 1, 3

[2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 2

[3] Tianqi Chen, Thierry Moreau, Ziheng Jiang, Lianmin Zheng, Eddie Yan, Haichen Shen, Meghan Cowan, Leyuan Wang, Yuwei Hu, Luis Ceze, et al. TVM: An automated End-to-End optimizing compiler for deep learning. In *OSDI*, pages 578–594, 2018. 1, 2, 3

[4] Xiangxiang Chu, Bo Zhang, and Ruijun Xu. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. In *ICCV*, pages 12239–12248, 2021. 3

[5] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 1

[6] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *ICCV*, 2019. 1, 3

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 1, 3

[8] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 1

[9] Sachin Mehta and Mohammad Rastegari. Mobilevit: Lightweight, general-purpose, and mobile-friendly vision transformer. In *ICLR*, 2021. 1, 3

[10] Pavlo Molchanov, Arun Mallya, Stephen Tyree, Iuri Frosio, and Jan Kautz. Importance estimation for neural network pruning. In *CVPR*, pages 11264–11272, 2019. 1

[11] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, pages 4510–4520, 2018. 3

[12] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *CVPR*, pages 2820–2828, 2019. 3

[13] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019. 3

[14] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*. PMLR, 2021. 1, 3

[15] Huanrui Yang, Hongxu Yin, Pavlo Molchanov, Hai Li, and Jan Kautz. Nvit: Vision transformer compression and parameter redistribution. *arXiv preprint arXiv:2110.04869*, 2021. 1