REPVIT: REVISITING MOBILE CNN FROM VIT PERSPECTIVE

Ao Wang[1,2], Hui Chen[1,2,†], Zijia Lin[1], Jungong Han[3], Guiguang Ding[1,2,†]
[1]Tsinghua University [2]BNRist [3]The University of Sheffield

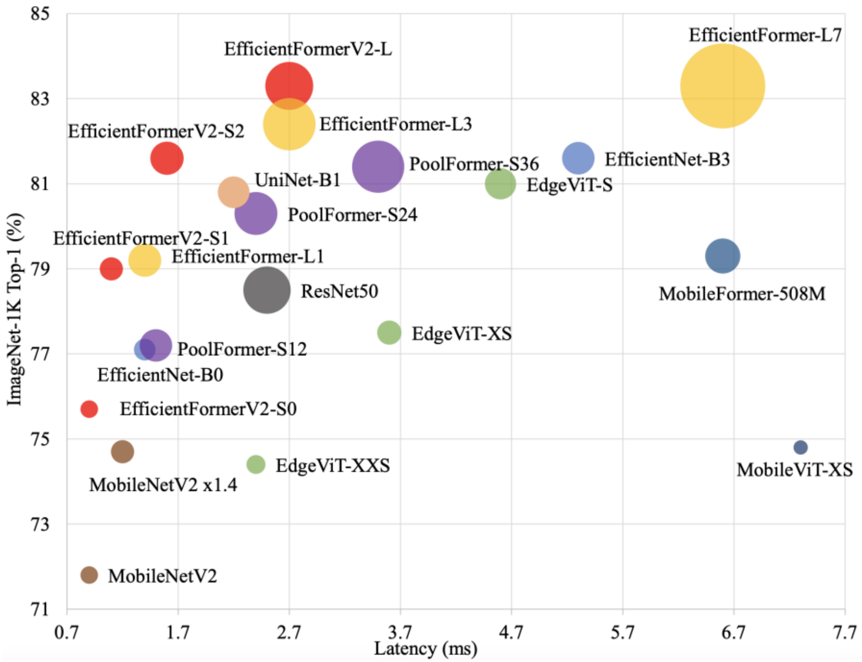
Background

Lightweight vision model architecture： CNN and ViT

- CNN architectures： MobileNets， MobileOne.
- ViT architectures： EfficientFormerV2， FastViT.

Previous optimization strategies for lightweight ViT：

- Introduce alternative attention mechanisms with lower complexity and less computations, e.g., MobileViTV2.
- Achieving hybrid architecture by incorporating convolutions, e.g., EfficientFormerV2.



Lightweight ViT exhibits superior performance and efficiency over lightweight CNN.

Background

Limitations of lightweight ViT

- Inadequate hardware and computational library support for attention on mobile and edge devices.
- Susceptible to inputs with high resolution, resulting in high latency.

Advantage of lightweight CNN

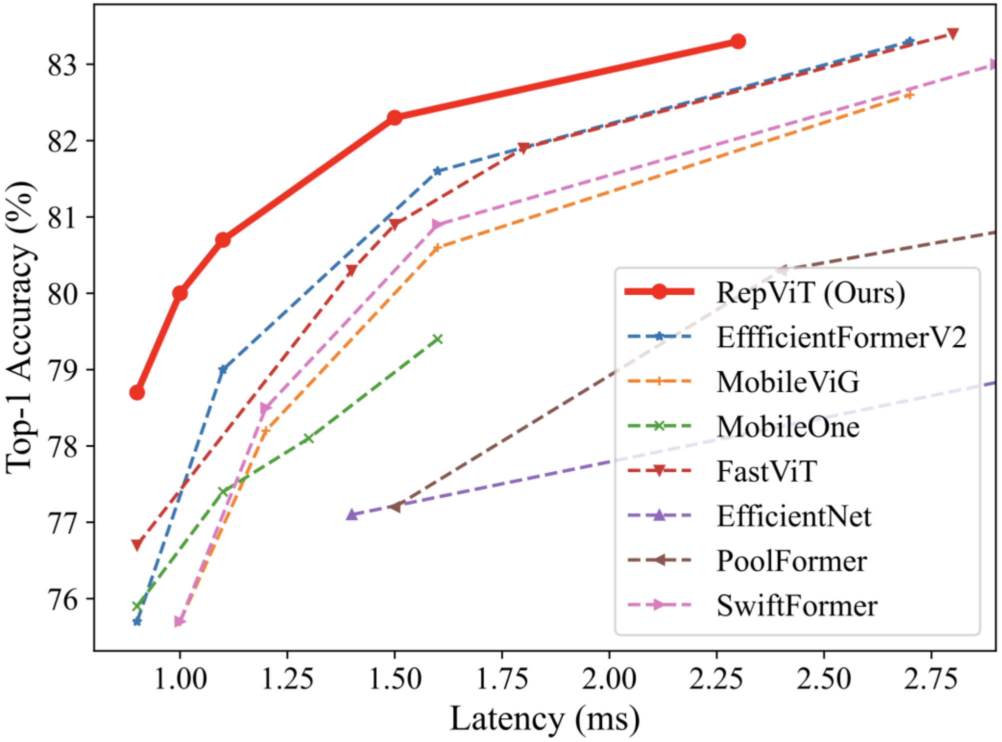
- Highly optimized and supported convolution operations on various hardwares.
- Linear complexity with respect to the input resolution, ensuring low latency.

Designing more efficient lightweight CNNs exhibits prospect for real-world scenarios.

Background

A new lightweight CNN: RepViT

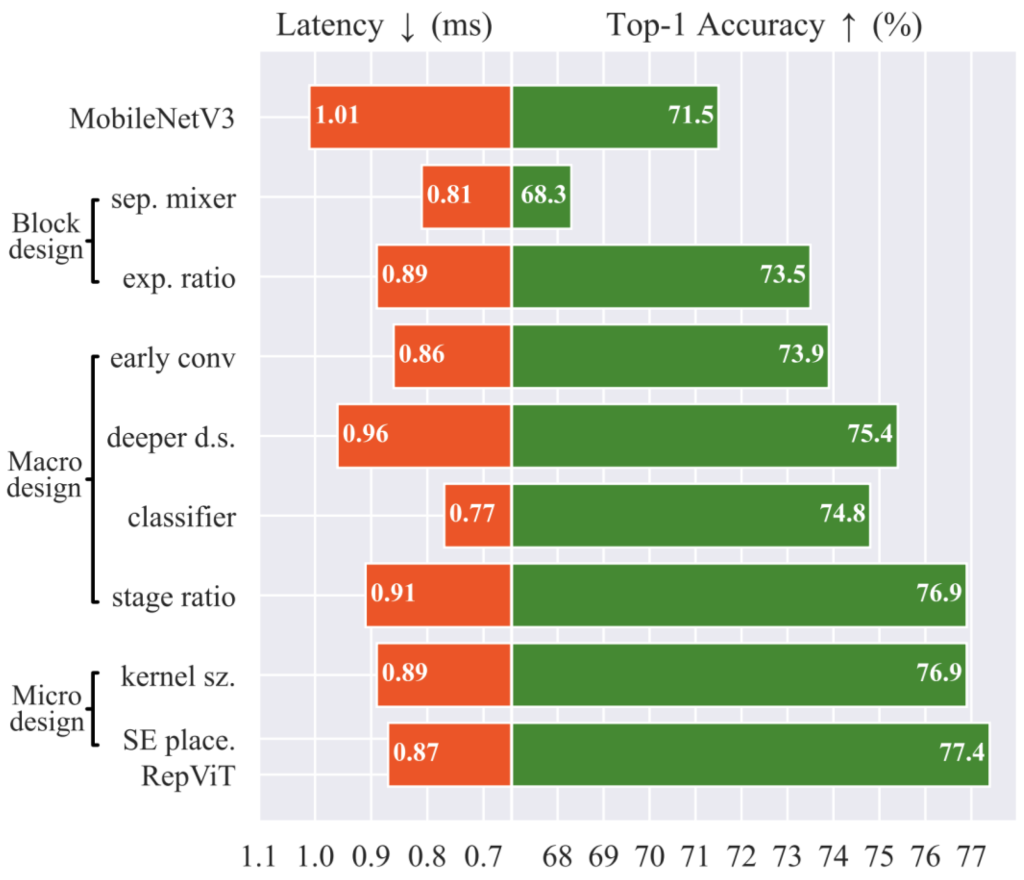
- Only employing convolutions, exhibiting low latency and high efficiency on mobile devices.
- Incorporate the effective architectural designs of lightweight ViTs, showing strong performance.
- Achieve over 80% top-1 accuracy with only 1.0 ms latency on an iPhone 12 for the first time.
- Explore the optimal block design, macro architectures, and micro architectures.



Methodology

RepViT: based on MobileNetV3-L with incorporating the effective lightweight ViT designs.

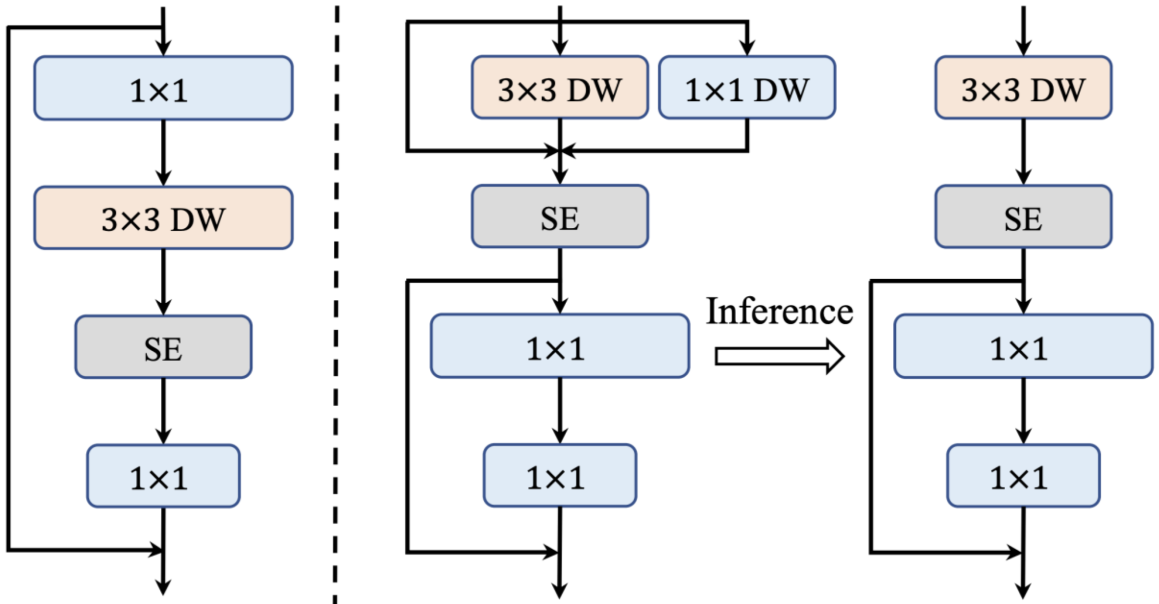
- Block design
  - From inverted residual block to MetaFormer structure.
- Macro design
  - For stem, downsampling, classifier.
  - Optimize the stage ratio for stages.
- Micro design
  - Investigate the kernel size selection.
  - Explore Squeeze-and-excitation layer placement.



Methodology

Block design

- Separate token mixer and channel mixer
  - Adopt the effective MetaFormer architecture.
  - Use structural reparam. for multi branch typology.
  - (3*3 DW + 1*1 DW)=>PW=>PW
- Reducing expansion ratio and increasing width
  - Mitigate the noticeable redundancy in FFN.
  - Increase the width to remedy the model capacity.



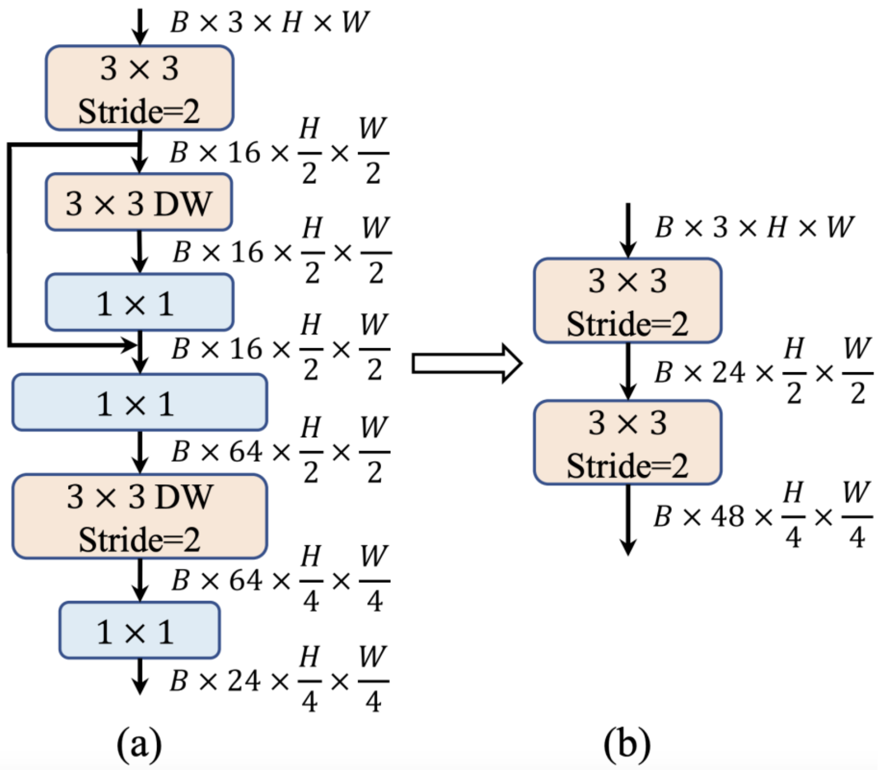
(a) MobileNet Block (b) RepViT Block

Latency： 1.01ms => 0.89ms
Accuracy： 71.5% => 73.5%

Methodology

Macro design for stem

- Stem has important impact on latency due to its processing on the feature map with the highest resolution.
- Early convolutions improve the optimization stability and performance for lightweight ViTs.
- Stem in MobileNetV3-L is complex, with latency bottleneck for mobile and weak capacity due to small width.
- Adopt early convolutions as stem with stacked stride-two convolutions to enhance both latency and accuracy



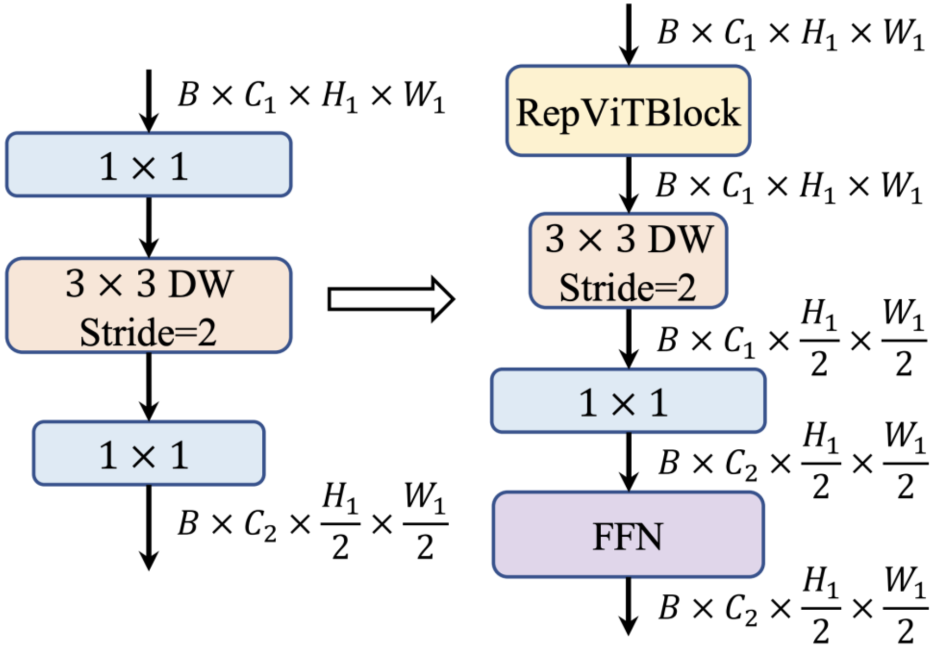
Latency： 0.89ms => 0.86ms
Accuracy： 73.5% => 73.9%

Methodology

Macro design for downsampling

- Lightweight ViTs employ separate downsampling layers to deepen the network depth and mitigate information loss.
- Downsampling in MobileNetV3 is coupled with the inverted residual block, bringing negative impact for performance.
- Separate the spatial reduction and channel modulation.
- Adopt RepViTBlock and FFN to deepen downsampling layers and enhance the performance under low cost.

Latency: 0.86ms => 0.96ms
Accuracy: 73.9% => 75.4%



(c) (d)

Methodology

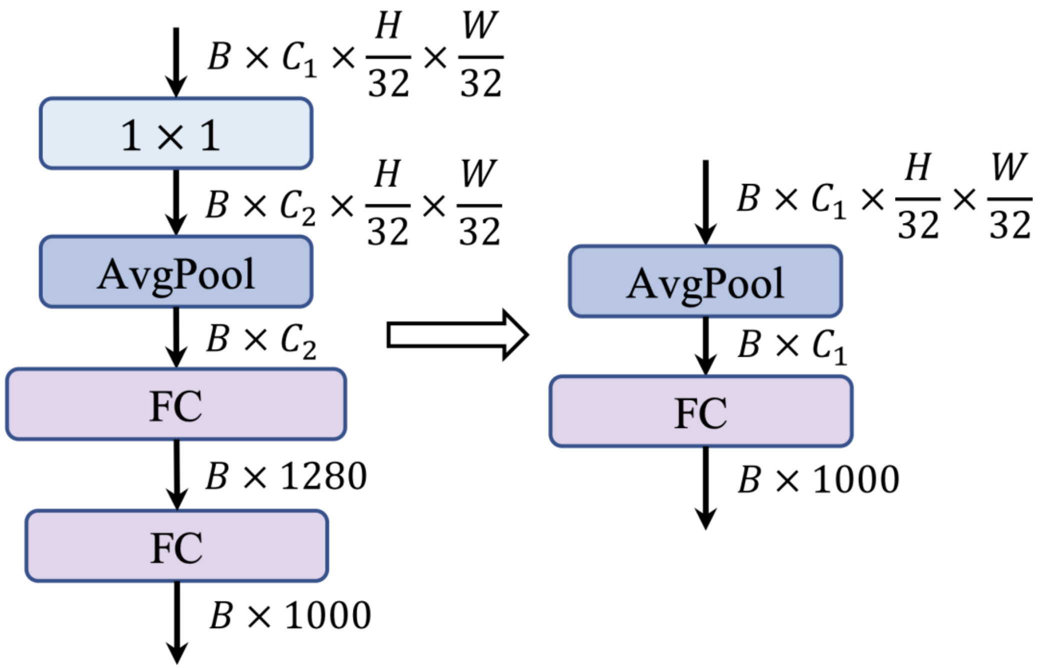
Macro design for classfier

- MobileNetV3 relies on the heavy classifier with high hidden dimensions for rich predictive features.
- The classifier processes the feature map in the largest dimension with big impact on latency.
- Adopt the simple classifier in lightweight ViT with global average pooling and linear layer for prediction.

Macro design for stage ratio

- More blocks in the third stage confer more favorable accuracy-latency balance.
- Employ the stage ratio of 1:1:7:1 like lightweight ViT.

Latency： 0.96ms => 0.77ms => 0.91ms
Accuracy： 75.4% => 74.8% => 76.9%



Methodology

Micro design

- Kernel size selection
  - Large kernel benefits the performance but lacks sufficient support on mobile devices and causes noticeable latency.
  - Adopt the highly optimized 3*3 convolutions for all blocks.
- Squeeze-and-excitation layer placement
  - Introduce data-driven attribute for convolutions by SE layer.
  - SE brings performance improvement but also with latency increase.
  - Utilize SE layer in the cross-block manner for maximal benefit

Latency： 0.91ms => 0.89ms => 0.87ms
Accuracy： 76.9% => 76.9% => 77.4%

Experiments

RepViT exhibits superior performance on ImageNet-1K

Model	Type	Params (M)	GMACs	Latency ↓ (ms)	Throughput ↑ (im/s)	Epochs	Top-1 (%)
MobileViG-Ti [48]	CNN-GNN	5.2	0.7	1.0	4337	300	75.7
FastViT-T8 [59]	Hybrid	3.6	0.7	0.9	3909	300	76.7
SwiftFormer-XS [55]	Hybrid	3.5	0.6	1.0	4304	300	75.7
EfficientFormerV2-S0 [35]	Hybrid	3.5	0.4	0.9	1274	300 / 450	75.7 / 76.2
RepViT-M0.9	CONV	5.1	0.8	**0.9**	4817	300 / 450	**78.7 / 79.1**
RepViT-M1.0	CONV	6.8	1.1	**1.0**	3910	300 / 450	**80.0 / 80.3**
MobileViG-S [48]	CNN-GNN	7.2	1.0	1.2	2985	300	78.2
EfficientFormer-L1 [36]	Hybrid	12.3	1.3	1.4	3360	300	79.2
SwiftFormer-S [55]	Hybrid	6.1	1.0	1.2	3376	300	78.5
EfficientFormerV2-S1 [35]	Hybrid	6.1	0.7	1.1	1153	300 / 450	79.0 / 79.7
RepViT-M1.1	CONV	8.2	1.3	**1.1**	3604	300 / 450	**80.7 / 81.2**
MobileViG-M [48]	CNN-GNN	14.0	1.5	1.6	2491	300	80.6
FastViT-S12 [59]	Hybrid	8.8	1.8	1.5	2313	300	80.9
FastViT-SA12 [59]	Hybrid	10.9	1.9	1.8	2181	300	81.9
SwiftFormer-L1 [55]	Hybrid	12.1	1.6	1.6	2576	300	80.9
EfficientFormerV2-S2 [35]	Hybrid	12.6	1.3	1.6	611	300 / 450	81.6 / 82.0
RepViT-M1.5	CONV	14.0	2.3	**1.5**	2151	300 / 450	**82.3 / 82.5**
MobileViG-B [48]	CNN-GNN	26.7	2.8	2.7	1446	300	82.6
EfficientFormer-L3 [36]	Hybrid	31.3	3.9	2.7	1422	300	82.4
EfficientFormer-L7 [36]	Hybrid	82.1	10.2	6.6	619	300	83.3
SwiftFormer-L3 [55]	Hybrid	28.5	4.0	2.9	1474	300	83.0
EfficientFormerV2-L [35]	Hybrid	26.1	2.6	2.7	399	300 / 450	83.3 / 83.5
RepViT-M2.3	CONV	22.9	4.5	**2.3**	1184	300 / 450	**83.3 / 83.7**

RepViT-M0.9 outperforms FastViT-T8 by 2%

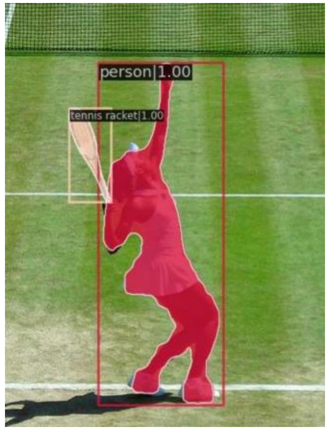
RepViT-M1.1 outperforms EfficientFormerV2-S1 by 1.7%

For the first time, RepViT-M1.0 achieves over 80% with 1ms latency on an iPhone12.

Experiments

RepViT exhibits advantages on downstream object detection and instance segmentation tasks.

Backbone	Latency ↓ (ms)	Object Detection			Instance Segmentation			Semantic
		AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	mIoU
ResNet18 [23]	4.4	34.0	54.0	36.7	31.2	51.0	32.7	32.9
PoolFormer-S12 [69]	7.5	37.3	59.0	40.1	34.6	55.8	36.9	37.2
EfficientFormer-L1 [36]	5.4	37.9	60.3	41.0	35.4	57.3	37.3	38.9
RepViT-M1.1	4.9	39.8	61.9	43.5	37.2	58.8	40.1	40.6
PoolFormer-S24 [69]	12.3	40.1	62.2	43.4	37.0	59.1	39.6	40.3
PVT-Small [63]	53.7	40.4	62.9	43.8	37.8	60.1	40.3	39.8
EfficientFormer-L3 [36]	12.4	41.4	63.9	44.7	38.1	61.0	40.4	43.5
RepViT-M1.5	6.4	41.6	63.2	45.3	38.6	60.5	41.5	43.6
EfficientFormerV2-S2* [35]	12.0	43.4	65.4	47.5	39.5	62.4	42.2	42.4
EfficientFormerV2-L* [35]	18.2	44.7	66.3	48.8	40.4	63.5	43.2	45.2
RepViT-M2.3*	9.9	44.6	66.1	48.8	40.8	63.6	43.9	46.1



RepViT-M1.1 outperforms EfficientFormer-L1 by 1.9 box AP and 1.8 mask AP.

RepViT-M2.3 is nearly 2x faster than EfficientFormerV2-L with the similar performance.

Experiments

RepViT can serve as the efficient image encoder for SAM

RepViT-SAM is nearly 10x faster than MobileSAM.

Platform	Image encoder			Mask decoder
	RepViT-SAM	MobileSAM [71]	ViT-B-SAM [33]	
iPhone	**48.9**	OOM	OOM	11.6
Macbook	**44.8**	482.2	6249.5	11.8

RepViT-SAM exhibits strong performance on zero-shot edge detection, instance segmentation, and segmentation in the wild tasks.

Model	z.s. edge.			z.s. ins.	SegInW
	ODS	OIS	AP	AP	Mean AP
ViT-H-SAM [33]	**.768**	**.786**	**.794**	**46.8**	**48.7**
ViT-B-SAM [33]	.743	.764	.726	42.5	44.8
MobileSAM [71]	.756	.768	.746	42.7	43.9
RepViT-SAM	.764	**.786**	.773	44.4	46.1

THANKS！