RePaViT: Scalable Vision Transformer Acceleration via Structural Reparameterization on Feedforward Network Layers

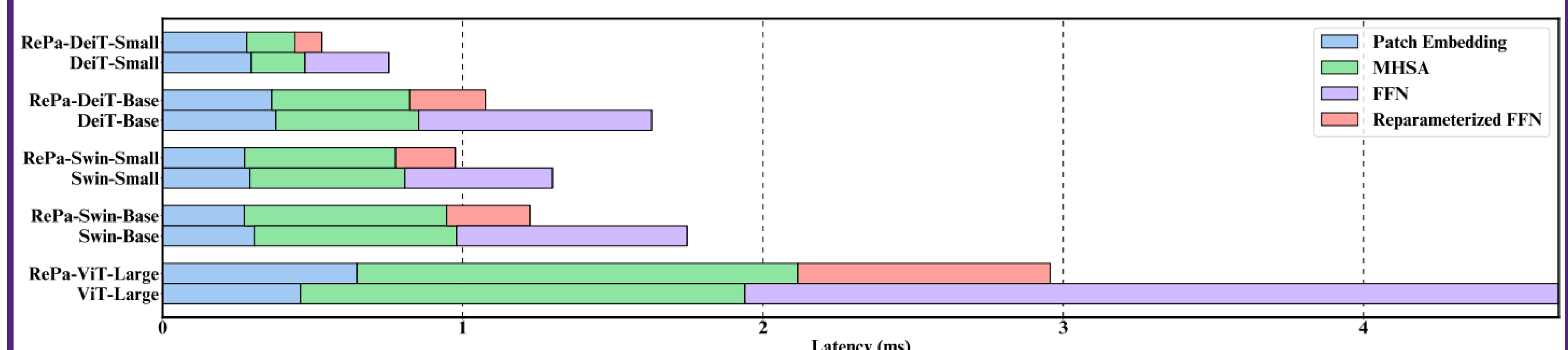
Xuwei Xu[1,2], Yang Li[3], Yudong Chen[1], Jiajun Liu[3,1], Sen Wang[1,2]

[1]The University of Queensland, Australia [2]ARC Training Centre for Information Resilience, Australia [3]CSIRO DATA61, Australia

Source code

Motivation

- Existing efficient ViT methods often **overlook the latency of FFN layers**, which can contribute to more than 60% of the inference latency in large-scale ViT models.
- The **proportion of FFN layers** in the total inference latency **escalates as model size increases**.

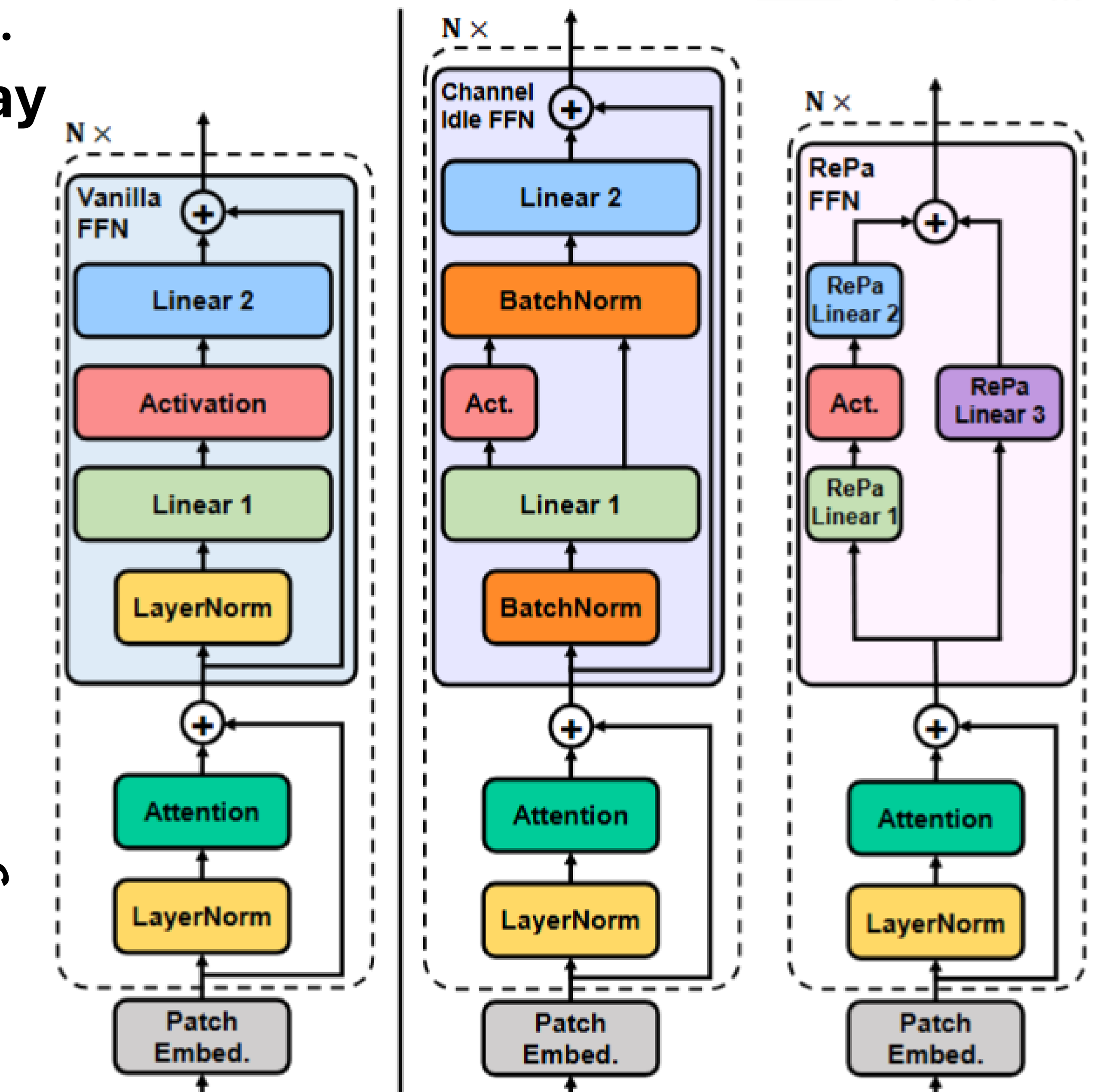


- **Structural reparameterization** technique can simplify neural networks by linear algebra operations. However, its effectiveness **on condensing FFN layers has barely been studied**.

Method

- To facilitate structural reparameterization on FFN layers, we keep some channels **idle** without being activated. As a result, these idle channels form a **linear pathway** through the activation function.
- **Vanilla FFN layers:** $O(2\rho NC^2)$
  1. $X^{\mathrm{In}} = \mathrm{LN}(X)W^{\mathrm{In}}$
  2. $X^{\mathrm{Act}} = \mathrm{Act}(X^{\mathrm{In}})$
  3. $Y = X^{\mathrm{Act}}W^{\mathrm{Out}} + X$
- **Ours during training:** $O(2\rho NC^2)$
  1. $X^{\mathrm{In}} = \mathrm{BN}(X)W^{\mathrm{In}}$
  2. $X^{\mathrm{Act}} = \mathrm{Act}(X^{\mathrm{In}}_{[:,\,:\mu C]}),\quad X^{\mathrm{Idle}} = X^{\mathrm{In}}_{[:,\,\mu C+1:]}$
  3. $X^{\mathrm{Con}} = \mathrm{Concat}(X^{\mathrm{Act}}, X^{\mathrm{Idle}})$
  4. $Y = \mathrm{BN}(X^{\mathrm{Con}})W^{\mathrm{Out}} + X$
- **Ours during inference:** $O\left((2\mu + 1)NC^2\right)$
  1. $Y = \mathrm{Act}(X\widetilde{W}^{\mathrm{In}})\widetilde{W}^{\mathrm{Out}} + X\widetilde{W}$

62.5% GFLOPs and #Params reduction for FFN layers



(a) ViT Block (b) RePaViT Block (c) Reparameterized RePaViT Block

Results

- RePaViT achieves more significant accelerations as model size increases. **68.7% faster inference** speed on ViT-Large.
- RePaViT realizes narrower performance gaps and even improves performance as model scales up. **1.7% higher accuracy** on ViT-Large.
- RePaViT works on various ViT backbones and has potentials on large foundation models.

Model	RePa	#MParam. ↓	Complexity (GMACs) ↓	Speed (images/second) ↑	Top-1 accuracy ↑
DeiT-Tiny	-	5.7	1.1	3435.1	72.1%
RePa-DeiT-Tiny/0.50	×	5.7	1.1	2397.9	69.4% (−2.7%)
	√	4.4 (−22.8%)	0.8 (−27.3%)	4001.2 (+16.5%)	
DeiT-Small	-	22.1	4.3	1410.3	79.8%
RePa-DeiT-Small/0.5	×	22.1	4.3	1000.9	78.9% (−0.9%)
	√	16.7 (−24.4%)	3.2 (−25.6%)	1734.7 (+23.0%)	
DeiT-Base	-	86.6	16.9	418.5	81.8%
RePa-DeiT-Base/0.75	×	86.6	16.9	336.6	81.3% (−0.5%)
	√	51.1 (−41.0%)	9.9 (−41.4%)	660.3 (+57.8%)	
ViT-Large	-	304.3	59.7	124.2	80.3%
RePa-ViT-Large/0.75	×	304.5	59.8	102.7	82.0% (+1.7%)
	√	178.4 (−41.4%)	34.9 (−41.5%)	207.2 (+66.8%)	
ViT-Huge	-	632.2	124.3	61.5	80.3%
RePa-ViT-Huge/0.75	×	632.5	124.4	53.0	81.4% (+1.1%)
	√	369.9 (−41.5%)	72.6 (−41.6%)	103.8 (+68.7%)	
Swin-Tiny	-	28.3	4.4	804.4	81.2%
RePa-Swin-Tiny/0.75	×	28.3	4.4	614.9	78.4% (−2.8%)
	√	17.5 (−38.2%)	2.6 (−40.9%)	1020.4 (+26.9%)	
Swin-Small	-	49.6	8.6	471.7	83.0%
RePa-Swin-Small/0.75	×	49.7	8.6	363.1	81.4% (−1.6%)
	√	29.9 (−39.7%)	5.1 (−40.7%)	627.8 (+33.1%)	
Swin-Base	-	87.8	15.2	326.6	83.5%
RePa-Swin-Base/0.75	×	87.9	15.2	249.4	82.6% (−0.9%)
	√	52.8 (−39.9%)	9.0 (−40.8%)	467.6 (+43.2%)	

More Results

Model	Idle ratio θ	#MParam. ↓	Complexity (GFLOPs) ↓	Speed (image/second) ↑	Top-1 accuracy ↑
CLIP-ViT-B/32	-	87.9	4.4	3860.2	57.1%
RePa-CLIP-ViT-B/32	0.50	66.6 (−24.2%)	3.4 (−22.7%)	4893.5 (+26.8%)	56.8% (−0.3%)
RePa-CLIP-ViT-B/32	0.75	52.4 (−40.4%)	2.6 (−40.9%)	5812.3 (+50.6%)	53.2% (−3.9%)
CLIP-ViT-B/16	-	86.2	17.6	824.2	62.7%
RePa-CLIP-ViT-B/16	0.50	64.9 (−24.7%)	13.4 (−23.9%)	1027.9 (+24.7%)	63.5% (+0.8%)
RePa-CLIP-ViT-B/16	0.75	50.8 (−41.1%)	10.6 (−39.8%)	1161.5 (+40.9%)	61.0% (−1.7%)

↑ Performance on CLIP Comparison with state-of-the-arts →

↓ Performance on downstream tasks

Model	RetinaNet							Mask R-CNN							UperNet	
	Latency (ms) ↓	AP↑	AP_{50}↑	AP_{75}↑	AP_S↑	AP_M↑	AP_L↑	Latency (ms) ↓	AP↑	AP_{50}↑	AP_{75}↑	AP_S↑	AP_M↑	AP_L↑	Latency (ms) ↓	mIoU↑
Swin-Small	61.7	37.2	56.9	39.6	**22.4**	40.5	49.4	62.5	**45.5**	**67.8**	**49.9**	**28.6**	**49.2**	**60.4**	36.3	**47.6**
RePa-Swin-Small	**53.8** (−12.8%)	**38.3**	**57.9**	**40.7**	21.8	**42.0**	**51.6**	**53.8** (−13.9%)	43.6	65.8	47.8	27.1	47.0	57.3	**32.1** (−11.6%)	45.7
Swin-Base	82.0	38.9	59.5	41.3	24.3	43.6	**54.4**	82.6	**45.8**	**67.6**	**50.3**	28.7	**48.9**	**61.7**	45.6	**48.1**
RePa-Swin-Base	**66.7** (−18.7%)	**39.8**	**60.0**	**42.1**	**25.3**	**43.7**	53.8	**69.4** (−16.0%)	44.8	67.0	49.4	**29.0**	48.5	58.4	**38.6** (−15.4%)	46.9

Backbone	Method	#MParam. ↓	Compl. (GMACs) ↓	Speed improv. ↑	Top-1 acc. ↑
DeiT-Small	WDPruning	13.3	2.6	+18.3%	78.4%
	X-pruner	-	2.4	-	78.9%
	DC-ViT	16.6	3.2	+20.0%	78.6%
	LPViT	22.1	**2.3**	+16.3%	**80.7%**
	RePaViT/0.50	16.7	3.2	+23.0%	78.9%
	RePaViT/0.75	**13.2**	2.5	**+42.1%**	77.0%
DeiT-Base	WDPruning	55.3	9.9	+18.2%	80.8%
	X-pruner	-	**8.5**	-	81.0%
	DC-ViT	65.1	12.7	+18.4%	81.3%
	LPViT	86.6	8.8	+18.8%	80.8%
	RePaViT/0.50	65.3	12.7	+28.6%	**81.4%**
	RePaViT/0.75	**51.1**	10.6	**+57.8%**	81.3%
Swin-Small	WDPruning	32.8	6.3	+15.3%	81.8%
	X-pruner	-	6.0	-	82.0%
	RePaViT/0.50	37.8	6.4	+20.7%	**82.8%**
	RePaViT/0.75	**29.9**	**5.1**	**+33.1%**	81.4%
Swin-Base	DC-ViT	66.4	11.5	+14.9%	**83.8%**
	LPViT	87.8	11.2	+8.9%	81.7%
	RePaViT/0.50	66.8	11.5	+19.6%	83.4%
	RePaViT/0.75	**52.8**	**9.0**	**+42.4%**	82.6%