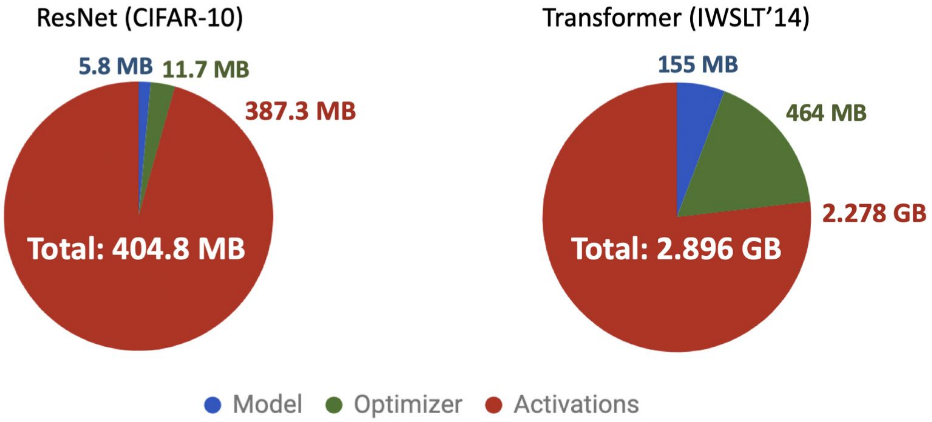
Reversible Vision Transformer
Mangalam et al., CVPR'22

Presenter: Kai Zhang
kaz321@lehigh.edu

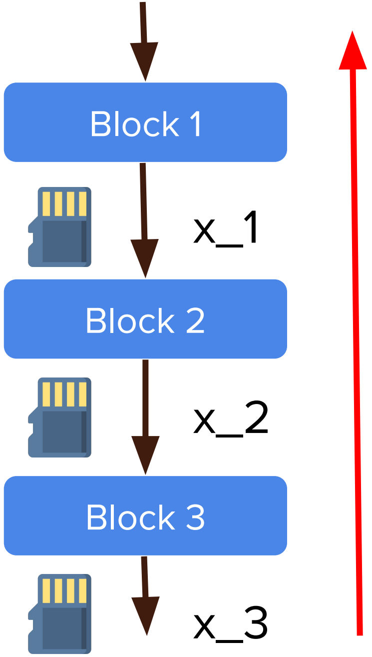
Motivation & Idea

TL;DR: Break **Memory Wall** via trading compute for memory through re-computation.

Memory Cost Analysis



$$\frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial \hat{y}} \left(\prod_{k=i+1}^{N} \frac{\partial f_k(x_k)}{\partial x_k} \right) \frac{\partial f_i(x_i)}{\partial w_i}$$



[1] Sohoni, Nimit Sharad, et al. "Low-memory neural network training: A technical report." arXiv preprint arXiv:1904.10631 (2019).

Backprop without Storing Activation

Each layer's activations can be **Reconstructed** exactly from next layer's.

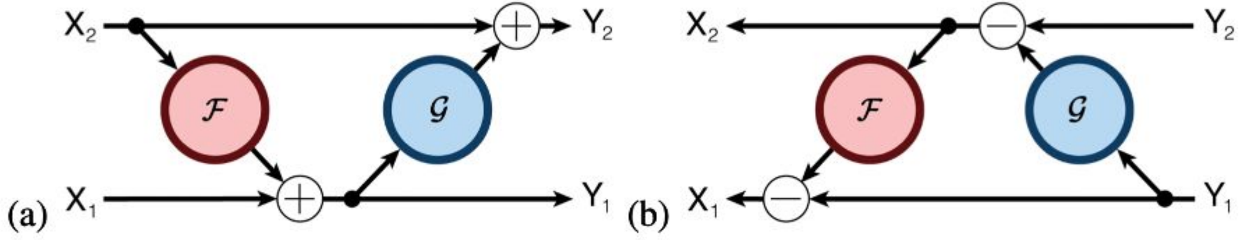


Figure 2: **(a)** the forward, and **(b)** the reverse computations of a residual block, as in Equation 8.

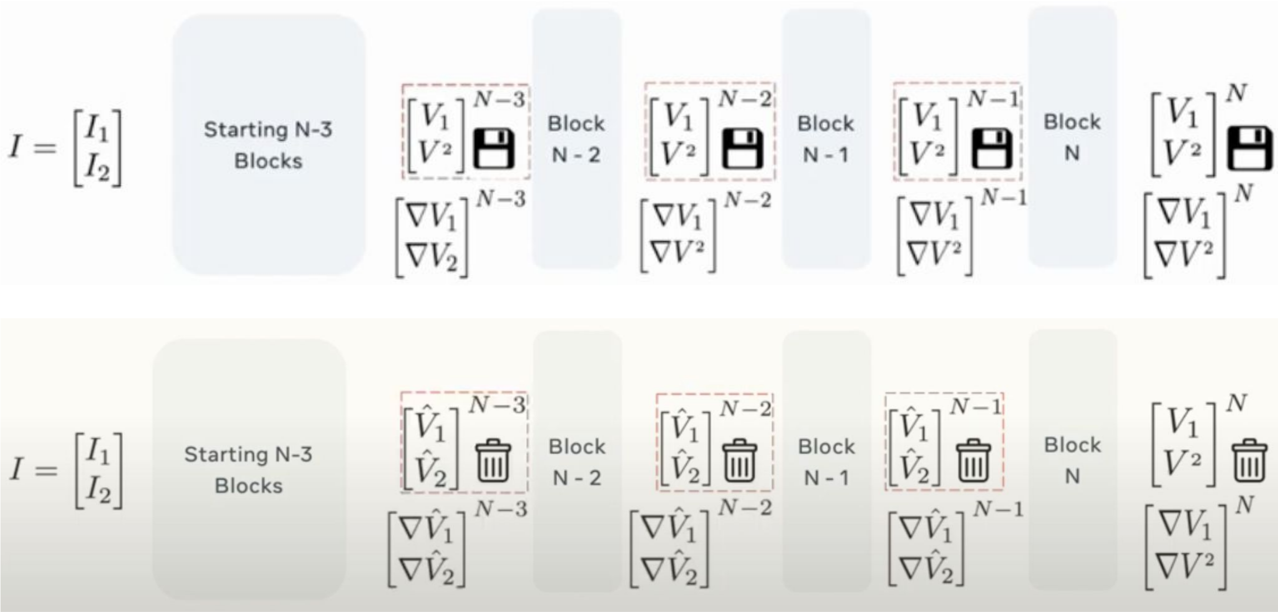
$$y_1 = x_1 + \boxed{\mathcal{F}}(x_2)$$
$$y_2 = x_2 + \mathcal{G}(y_1)$$

non-invertible

$$x_2 = y_2 - \mathcal{G}(y_1)$$
$$x_1 = y_1 - \mathcal{F}(x_2)$$

[2] N. Gomez et al. "The Reversible Residual Network: Backpropagation Without Storing Activations." NIPS (2017).

Vanilla *v.s.* Reversible Backprop



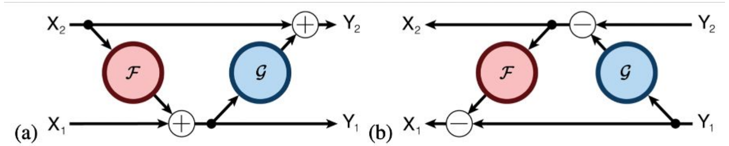
Activation Caching

Without Caching

Reversible Transforms in ViT

TL;DR: Adapting ViT to **Two-Residual-Streams**.

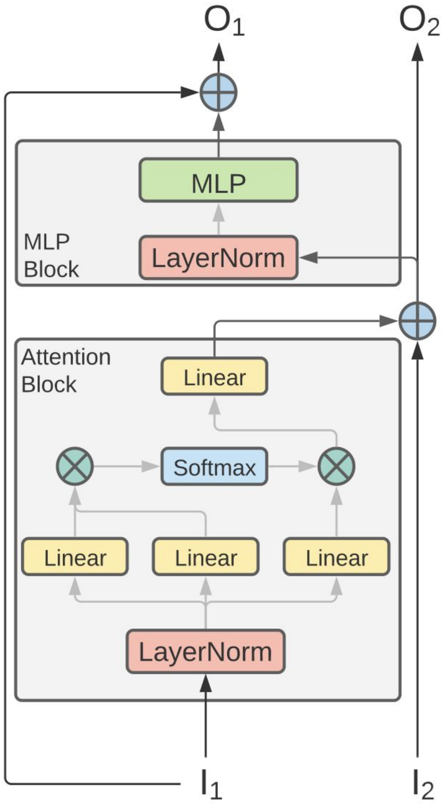
Rev-ViT Block



MLP

$$\mathbf{I} = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix} \xrightarrow[T]{} \begin{bmatrix} O_1 \\ O_2 \end{bmatrix} = \begin{bmatrix} I_1 + \boxed{G}(I_2 + F(I_1)) \\ I_2 + \boxed{F}(I_1) \end{bmatrix} = \mathbf{O}$$

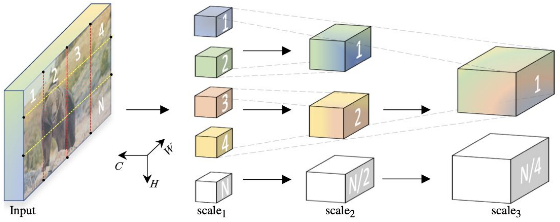
Multi-Head Attention



Note that Rev-ViT keep the patchification stem intact, while RevNet splits in halves along the channel dimensions.

<u>Dimension change hinders the reversible transformation.</u>

Multiscale Vision Transformer (Fan et al., CVPR'21)

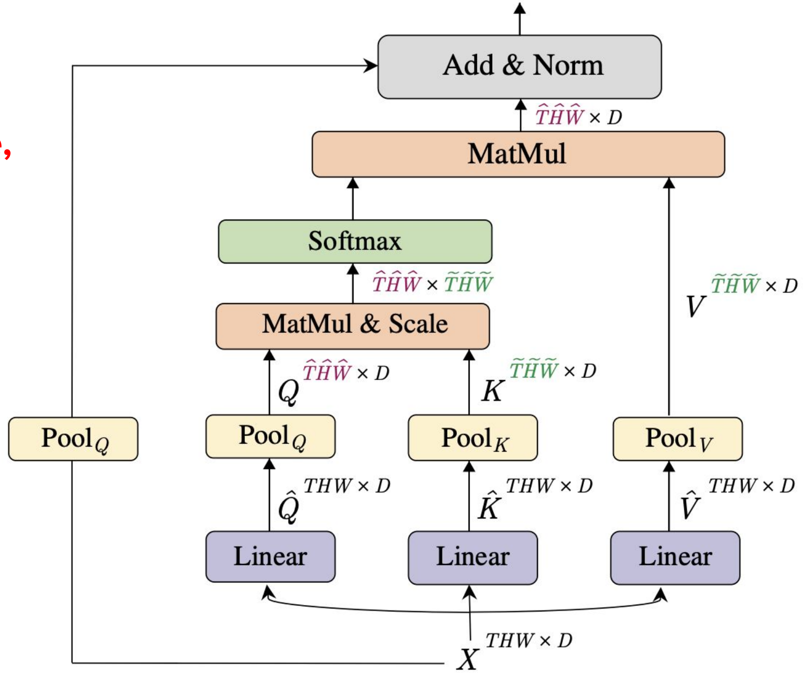


Pooling:

shorten sequence, enlarge channel.

stages	operators	output sizes
data layer	stride $\tau \times 1 \times 1$	$D \times T \times H \times W$
cube$_1$	$c_T \times c_H \times c_W$, D stride $s_T \times 4 \times 4$	$D \times \frac{T}{s_T} \times \frac{H}{4} \times \frac{W}{4}$
scale$_2$	$\begin{bmatrix} \text{MHPA}(D) \\ \text{MLP}(4D) \end{bmatrix} \times N_2$	$D \times \frac{T}{s_T} \times \frac{H}{4} \times \frac{W}{4}$
scale$_3$	$\begin{bmatrix} \text{MHPA}(2D) \\ \text{MLP}(8D) \end{bmatrix} \times N_3$	$2D \times \frac{T}{s_T} \times \frac{H}{8} \times \frac{W}{8}$
scale$_4$	$\begin{bmatrix} \text{MHPA}(4D) \\ \text{MLP}(16D) \end{bmatrix} \times N_4$	$4D \times \frac{T}{s_T} \times \frac{H}{16} \times \frac{W}{16}$
scale$_5$	$\begin{bmatrix} \text{MHPA}(8D) \\ \text{MLP}(32D) \end{bmatrix} \times N_5$	$8D \times \frac{T}{s_T} \times \frac{H}{32} \times \frac{W}{32}$

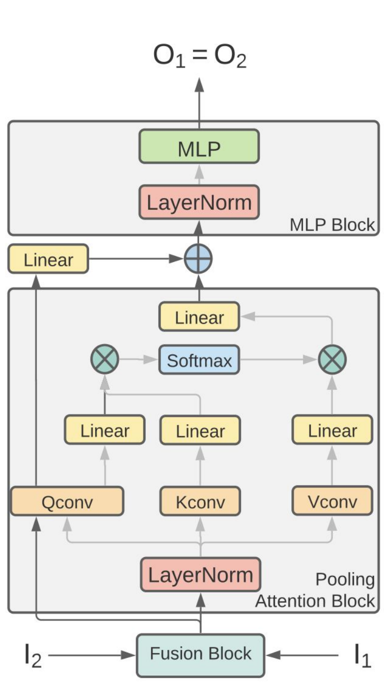
For each scale transition, the first MHPA layer does pooling, and the final MLP layer does upsampling.



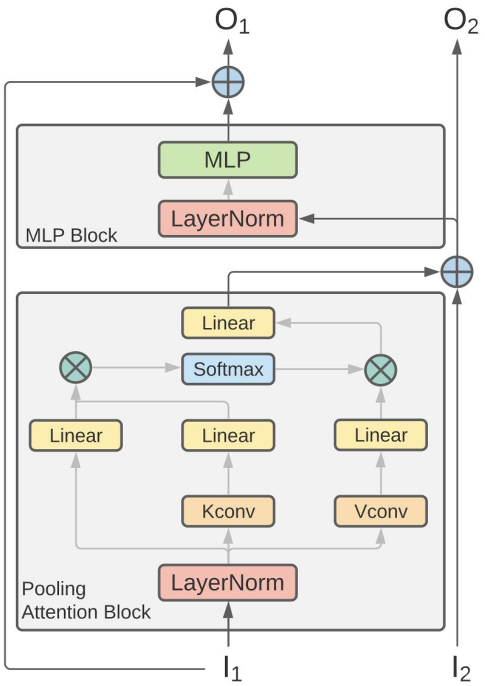
Rev-MViT

stage	operators		output sizes
data			224×224
cubification	7×7, 96 stride 4×4		96×56×56
Stage-Preserving	**F** : MHPA(96) **G** : MLP(384)	×1	Y_1 : 96×56×56 Y_2 : 96×56×56
Stage-Transition	FUSION(192) MHPA(192) MLP(768)	×1	192×28×28
Stage-Preserving	**F** : MHPA(192) **G** : MLP(768)	×1	Y_1 : 192×28×28 Y_2 : 192×28×28
Stage-Transition	FUSION(384) MHPA(384) MLP(1536)	×1	384×14×14
Stage-Preserving	**F** : MHPA(384) **G** : MLP(1536)	×10	Y_1 : 384×14×14 Y_2 : 384×14×14
Stage-Transition	FUSION(768) MHPA(768) MLP(3072)	×1	768×7×7
Stage-Preserving	**F** : MHPA(768) **G** : MLP(3072)	×1	Y_1 : 768×7×7 Y_2 : 768×7×7

Not reversible, we have to cache the activation.



(b) Stage-Transition Rev-**MViT** Block



(c) Stage-Preserving Rev-**MViT** Block

Experimental Results

TL;DR: Performance, Memory Efficiency, Speed.

Results

ImageNet-1K Classification

model	Acc	Memory (MB/img)	Maxiumum Batch Size	GFLOPs	Param (M)
ResNet-101 [29]	76.4	118.7	112	7.6	45
ResNet-152 [29]	77.0	165.2	79	11.3	60
RegNetY-4GF [58]	80.0	101.1	136	4.0	21
RegNetY-12GF [58]	80.3	175.2	75	12.1	51.8
RegNetY-32GF [58]	80.9	250.2	46	32.3	32.3
Swin-T [48]	81.3	-	-	4.5	29
ViT-S [63]	79.9	66.5	207	4.6	22
Rev-ViT-S	79.9	**8.8** ↓7.5×	**1232** ↑5.9×	4.6	22
ViT-B [63]	81.8	129.7	95	17.6	87
Rev-ViT-B	81.8	**17.0** ↓7.6×	**602** ↑6.3×	17.6	87
RegNetY-8GF [58]	81.7	147.2	91	8.0	39
CSWin-T [14]	82.7	-	-	4.3	23
Swin-S [48]	83.0	-	-	8.7	50
ViT-L	81.5	349.3	26	61.6	305
Rev-ViT-L	81.4	**22.6** ↓15.5×	**341** ↑13.1×	61.6	305
MViT-B-16 [18]	82.8	153.6	89	7.8	37
Rev-MViT-B-16	82.5	**66.8** ↓2.3×	**157** ↑1.8×	8.7	39

Kinetics-400 Video Classification.

model	top-1	**Mem** (GB)	**Max BS**	GFLOPs× views	Param
Two-Stream I3D [5]	71.6	-	-	216 × NA	25.0
R(2+1)D [66]	72.0	-	-	152×115	63.6
Two-Stream R(2+1)D [66]	73.9	-	-	304 × 115	127.2
Oct-I3D + NL [8]	75.7	-	-	28.9×3×10	33.6
ip-CSN-152 [65]	77.8	-	-	109×3×10	32.8
SlowFast 4×16, R50 [19]	75.6	-	-	36.1 × 30	34.4
SlowFast 8×8, R101 [19]	77.9	-	-	106 × 30	53.7
SlowFast 8×8 +NL [19]	78.7	-	-	116×3×10	59.9
ViT-B-VTN-IN-1K [52]	75.6	-	-	4218×1×1	114.0
ViT-B-VTN-IN-21K [52]	78.6	-	-	4218×1×1	114.0
MViT-B-16 , 16×4	78.4	1.27	10	70.5×1×5	36.6
Rev-MViT-B-16, 16×4	78.5	**0.64**	**20**	64×1×5	34.9

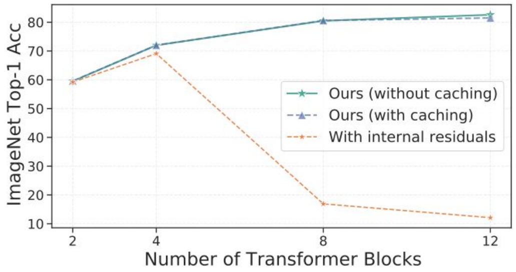
MSCOCO Object Detection

Model	AP^{box}	AP^{mask}	**Memory**(GB)	GFLOPs	Param (M)
Res50 [28]	41.0	37.1	-	260	44
Res101 [28]	42.8	38.5	-	336	63
X101-64 [73]	44.4	39.7	-	493	101
PVT-L [69]	44.5	40.7	-	364	81
MViT-B	48.2	43.9	18.9	668	57
Rev-MViT-B	48.0	43.5	10.9	683	58

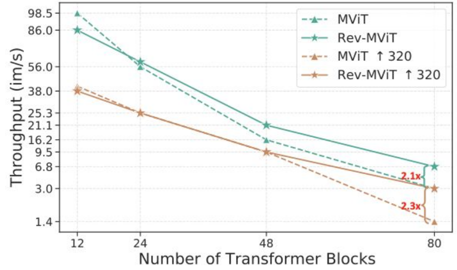
Ablation Study

The reversible models tend to have **stronger inherent regularization** than their non-reversible counterparts.

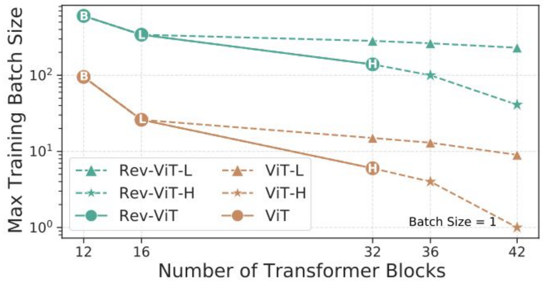
Training Improvement	Train Acc	Top-1 ImageNet Acc
Naïve Rev-ViT-B	15.3	12.1
+ Re-configuring residual streams	82.1	77.2
+ Repeated Augmentation	84.9	80.6
+ Lighter Augmentation magnitude	93.2	81.0
+ Stronger Stochastic Depth	92.0	81.4
+ Higher weight decay	91.0	81.8
Rev-ViT-B	91.0	81.8



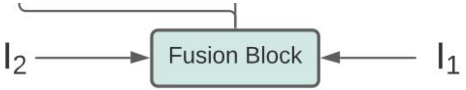
(a) Activation caching and internal residuals.



(b) Training throughput vs. Model Depth



(c) Reversible training and maximum batch size.



(b) Stage-Transition Rev-**MViT** Block

Stage-Transition Fusion	Termination Fusion	Train Acc	Top-1 Acc
2×-MLP	Norm → Concat	80.1	82.5

allows **efficient computation** of the resolution downsampling and feature upsampling without repeat computation in each stream separately.

LEHIGH
UNIVERSITY