TOKEN MERGING: YOUR VIT BUT FASTER

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang
Christoph Feichtenhofer, Judy Hoffman

Present By Louie Lu, Michael Tsai

Motivation

Motivation

- We want to improve the <u>**throughput**</u> of ViT models.
  - i.e. Improve model inference <u>**image/s**</u> (e.g. 100 im/s -> 200 im/s)
- Let's say we want to:
  - **2 times faster** than SotA
  - **Hotplug** into the model without re-training
  - **Without dropping** the accuracy
- How to do that?

How To Do That? Without Training & Improve Throughput?

Any thoughts?

How To Do That? Without Training & Improve Throughput?

- Having a more efficient transformer
- Token reduction by pruning, or masking
- Combining tokens

Introducing Token Merging - ToMe

Token Merging - ToMe

- Core Concept
  - ToMe **merges (combine)** the redundant tokens in each ViT layer.

Token Merging - ToMe

- Core Concept
  - ToMe **merges (combine)** the redundant tokens in each ViT layer.

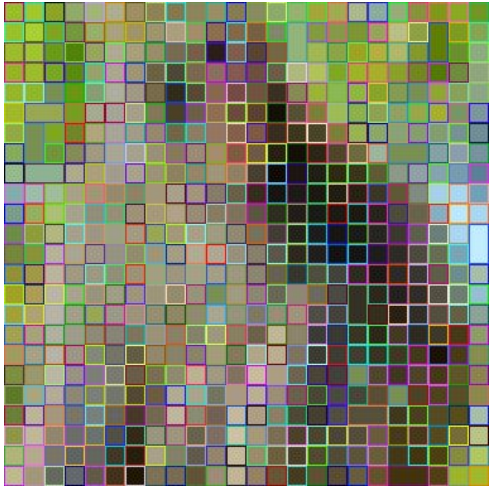
Taiwanese macaque, Kaohsiung, Taiwan. By Louie Lu,

Token Merging - ToMe

- Core Concept
  - ToMe **merges (combine)** the redundant tokens in each ViT layer.

Taiwanese macaque, Kaohsiung, Taiwan. By Louie Lu, all rights reserved.

Patchify



Token Merging - ToMe

- Core Concept
  - ToMe **merges (combine)** the redundant tokens in each ViT layer.

Block 8

Taiwanese macaque, Kaohsiung, Taiwan. B

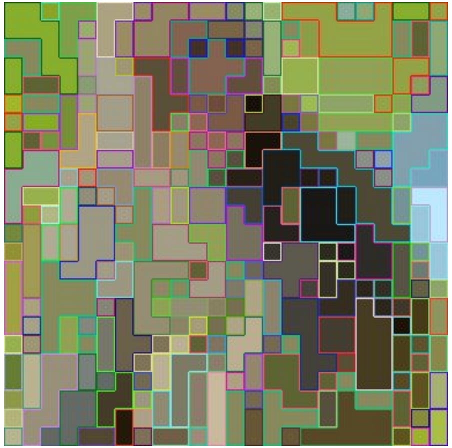
Token Merging - ToMe

- Core Concept
  - ToMe **merges (combine)** the redundant tokens in each ViT layer.

Taiwanese macaque, Kaohsiung, Taiwan. B
Louie Lu, all rights reserved.

Block 8

Block 16



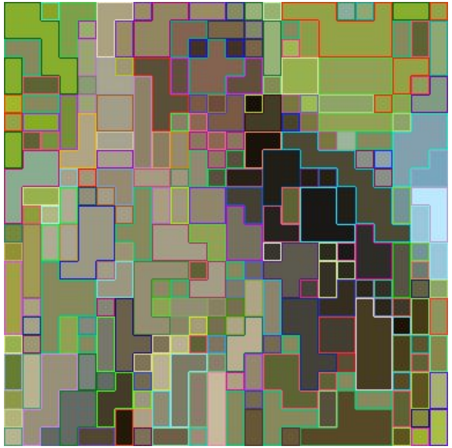
Token Merging - ToMe

Taiwanese macaque, Kaohsiung, Taiwan.

- Core Concept
  - ToMe **merges (combine)** the redundant tokens in each ViT layer.

Block 8

Block 16



Block 22

Token Merging - ToMe

- Core Concept
  - ToMe **merges (combine)** the redundant tokens in each ViT layer.
- When To Merge
  - Between the attention and MLP branches of each transformer block.
  - This enables on existing model without training!

Taiwanese macaque, Kaohsiung, Taiwan. By Louie Lu, all rights reserved.

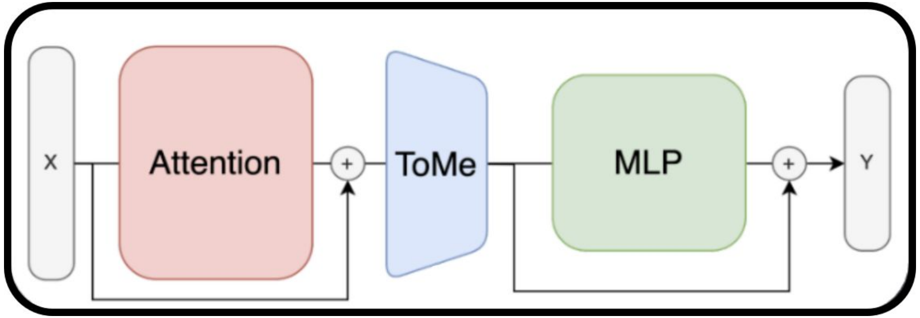


Fig. 1 (b), from ToMe, modified by Louie Lu. Declare fair use.

Token Merging - ToMe

- Core Concept
  - ToMe **merges (combine)** the redundant tokens in each ViT layer.
- When To Merge
  - Between the attention and MLP branches of each transformer block.
- Result
  - Image Classification: **2 times** faster than ViT-L @ 512
  - Video Classification: **2.2 times** faster than ViT-L

Taiwanese macaque, Kaohsiung, Taiwan. By Louie Lu, all rights reserved.

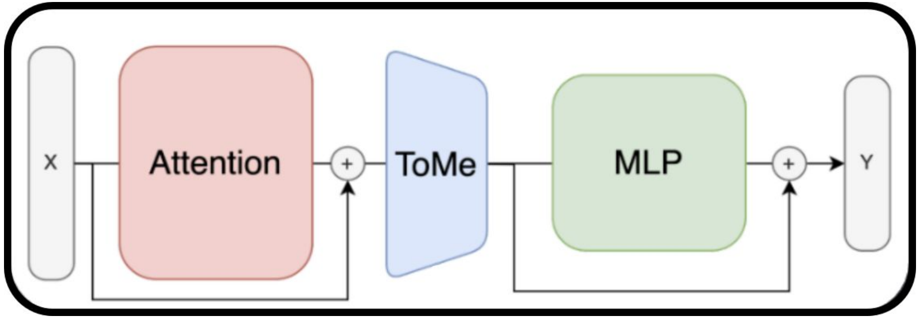


Fig. 1 (b), from [ToMe](), modified by Louie Lu. Declare fair use.

Token Merging - ToMe

- Technical Simplicity
  - Only 8 files, and **494 lines** of code.

Token Merging - ToMe

- Technical Simplicity
  - Only 8 files, and **494 lines** of code.
  - Easy to apply on existing model.
    - tome.patch.timm(model, trace_source=True)

Space High-level Summary of **To**ken **Me**rging - ToMe

- ToMe combines **similar tokens** in each layer.
- **It can hotplug to existing model.**
  - Without training, ToMe can hotplug to existing ViT model, and gain 2x throughput improvement without losing accuracy.
- **It can also applied on training**
  - ToMe shows 2x training speed on MAE fine-tuning on video.

Details of ToMe

Recap the Core of ToMe

- ToMe combines **similar tokens** in each layer.

Recap the Core of ToMe

- ToMe combines **similar tokens** in each layer.
- This yields the following questions:
  - What is similarity?
  - How many tokens combined in each layer?
  - How to combine?
  - How to maintain softmax attention result?
  - How to use ToMe in training?

1. Token Similarity

- Distance between two tokens in feature space?
  - Not necessarily optimal.
- Using transformer QKV self-attention keys (K)
  - Recap that the keys are already used in dot product similarity.
  - We define the token similarity as the dot product similarity metric (i.e. cosine similarity) between the keys of each token.

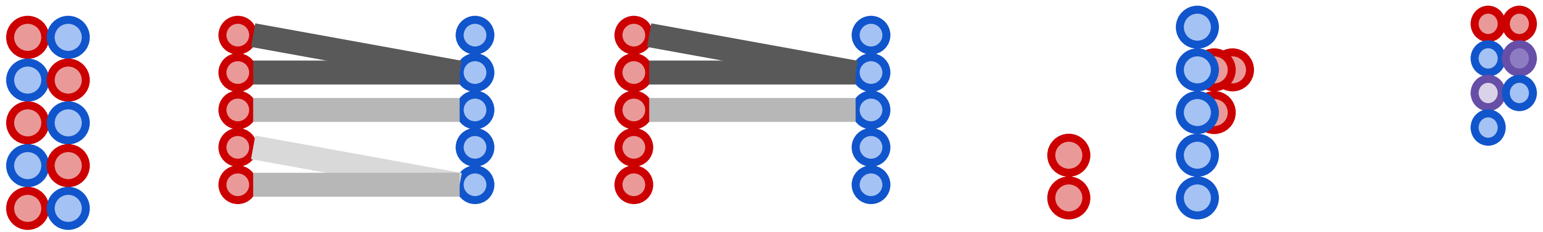
2. Token Combination Strategy

- We merge and reduce ***r*** tokens per block.
- For a model with ***L*** blocks, we gradually merge ***r * L*** tokens.
- ToMe **at most reduce 50%** of tokens.

2. Token Combination Strategy

- We merge and reduce ***r*** tokens per block.
- For a model with ***L*** blocks, we gradually merge ***r * L*** tokens.
- ToMe **at most reduce 50%** of tokens.
- E.g. Input size=**24x24+1**, ***L=24*** blocks model, and ***r=25*** ToMe
    - Patchify: 577 tokens (24x24+1 CLS token)
    - Block 1: 552 tokens left (+1 CLS token)
    - Block 22: 27 tokens left
    - Block 23: 14 tokens left (50% of 26 => reduce 13 tokens)
    - Block 24: 8 tokens left (50% of 13 => reduce 6 tokens)

3. Bipartite Soft Matching - In 5 Steps



Step 1. Assign Tokens to **Set A** and ***Set B***

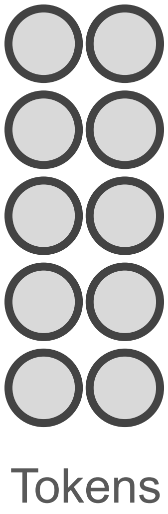
Step 2. Draw one edge from **Set A** token to their most similar token in ***Set B***

Step 3. Keep the top ***r*** most similar edges (e.g. ***r=3***)

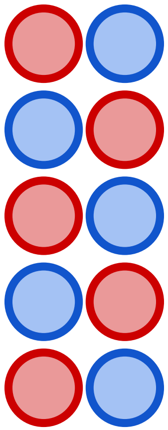
Step 4. Merge connected tokens

Step 5. Regroup the sets back together

3. Bipartite Soft Matching - In 5 Steps

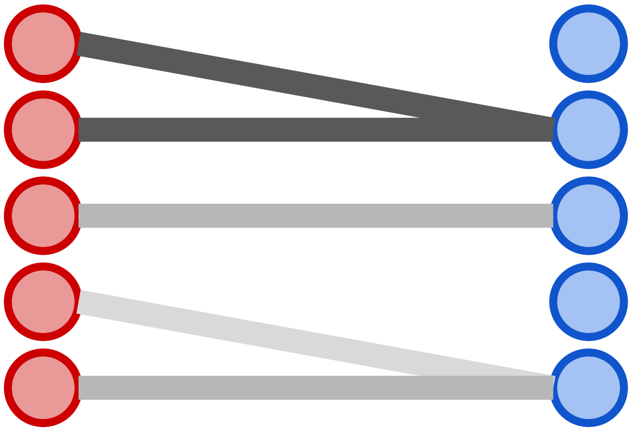


3. Bipartite Soft Matching - Step 1



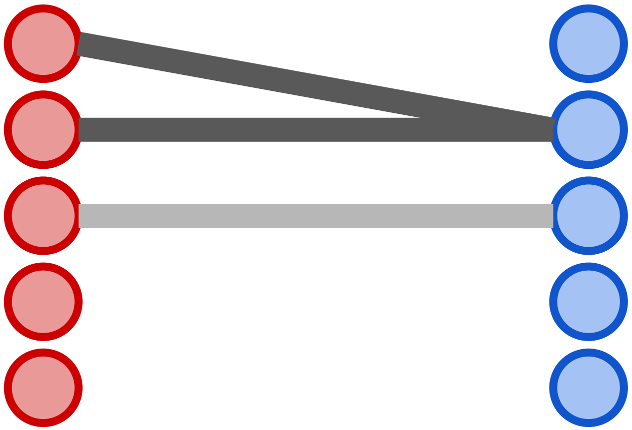
Assign Tokens to
Set A and ***Set B***

3. Bipartite Soft Matching - Step 2



Draw one edge from **Set A** token to their most similar token in ***Set B***

3. Bipartite Soft Matching - Step 3

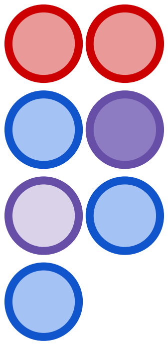


Keep the top ***r*** most similar edges
(e.g. ***r=3***)

3. Bipartite Soft Matching - Step 4

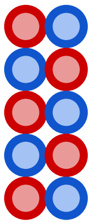
Merge connected tokens (e.g. averaging their features)

3. Bipartite Soft Matching - Step 5

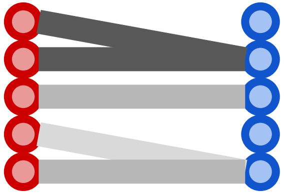


Regroup the sets back together

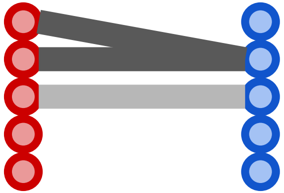
3. Bipartite Soft Matching - In 5 Steps



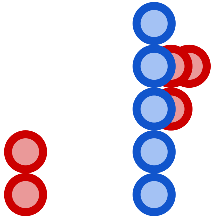
Step 1. Assign Tokens to **Set A** and ***Set B***



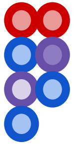
Step 2. Draw one edge from **Set A** token to their most similar token in ***Set B***



Step 3. Keep the top ***r*** most similar edges (e.g. ***r=3***)



Step 4. Merge connected tokens



Step 5. Regroup the sets back together

4. Proportional Attention

- If we merge the tokens with same key, it will changes the outcome of softmax attention.
- That is, that key has less effect in the softmax term.
- Fix by proportional attention:

$$\boldsymbol{A} = \text{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}} + \log \boldsymbol{s}\right)$$

  - Where ***s*** is the size of each token. (how many patches the token represents)

5. Training with ToMe

- Simply treat token merging as a pooling operation and backprop through the merged tokens as we were using average pooling.

Recap Details of ToMe

- Token Similarity
  - Use QKV self-attention keys **cosine similarity.**
- Token Combination Strategy
  - Reduce ***r*** tokens in each block, **at most reduce 50%** of tokens in each block.
- Bipartite Soft Matching
  - Linear merging on set A and B, average, and concatenate.
- Proportional Attention
  - Record **s=size of token**, add **log(s)** at attention
- When used in training, treat as pooling operation.

Ablation study

What do we want to know?

1.
2.
3.
4.

What to measure?

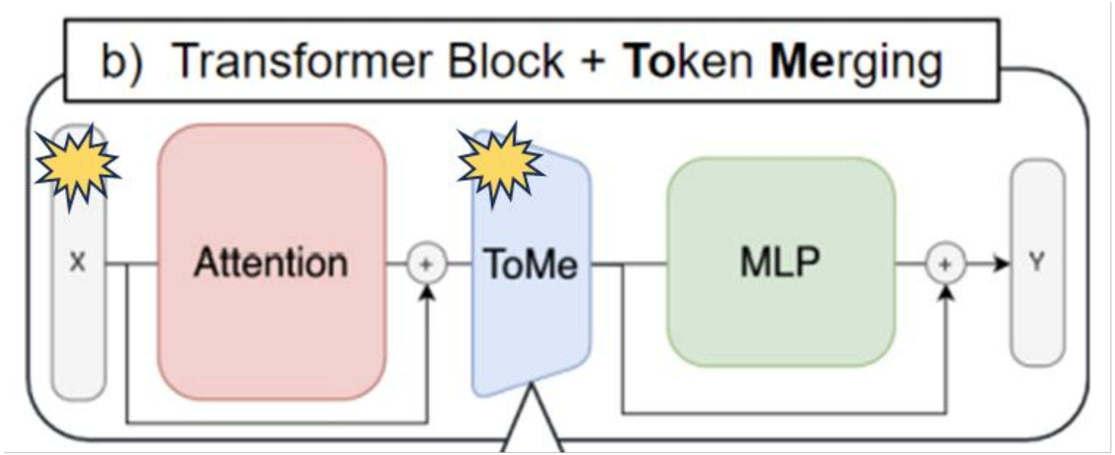
1.
2.

What do we want to know?

1. The input to the proposed module
2. Similarity method
3. Merging method
4. Group assignment method

What to measure?

1. Model performance
2. Speed



feature	acc	im/s
X_{pre}	83.02	**186.8**
X	83.70	182.8
K	**84.25**	182.9
Q	84.04	182.8
V	83.80	182.9

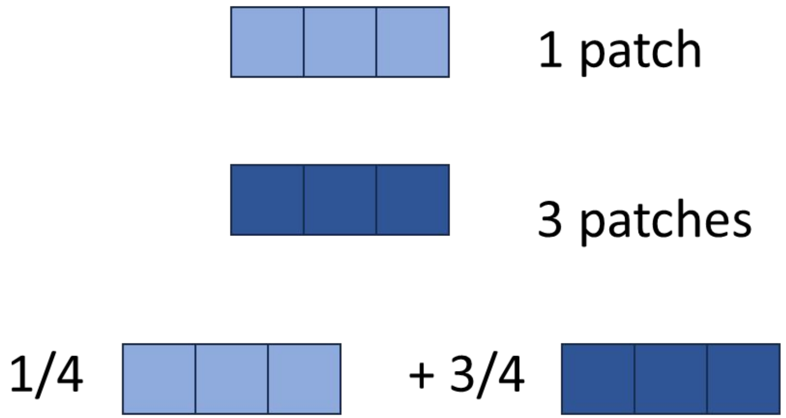
Similarity score

function	acc	im/s
eucl	**84.26**	182.5
cosine	**84.25**	**182.9**
dot	82.78	**183.0**
softmax	82.00	**183.0**

Merging method

method	acc	im/s
keep one	81.01	**185.4**
max pool	83.50	184.6
avg pool	83.57	183.8
weighted avg	**84.25**	182.9

(d) **Combining Method.** Averaging tokens weighted by their size, $\boldsymbol{s}$ (see Eq. 1), ensures consistency.



Group assignment method

order	acc	im/s
sequential	81.07	**183.0**
alternating	**84.25**	**182.9**
random	83.80	181.7

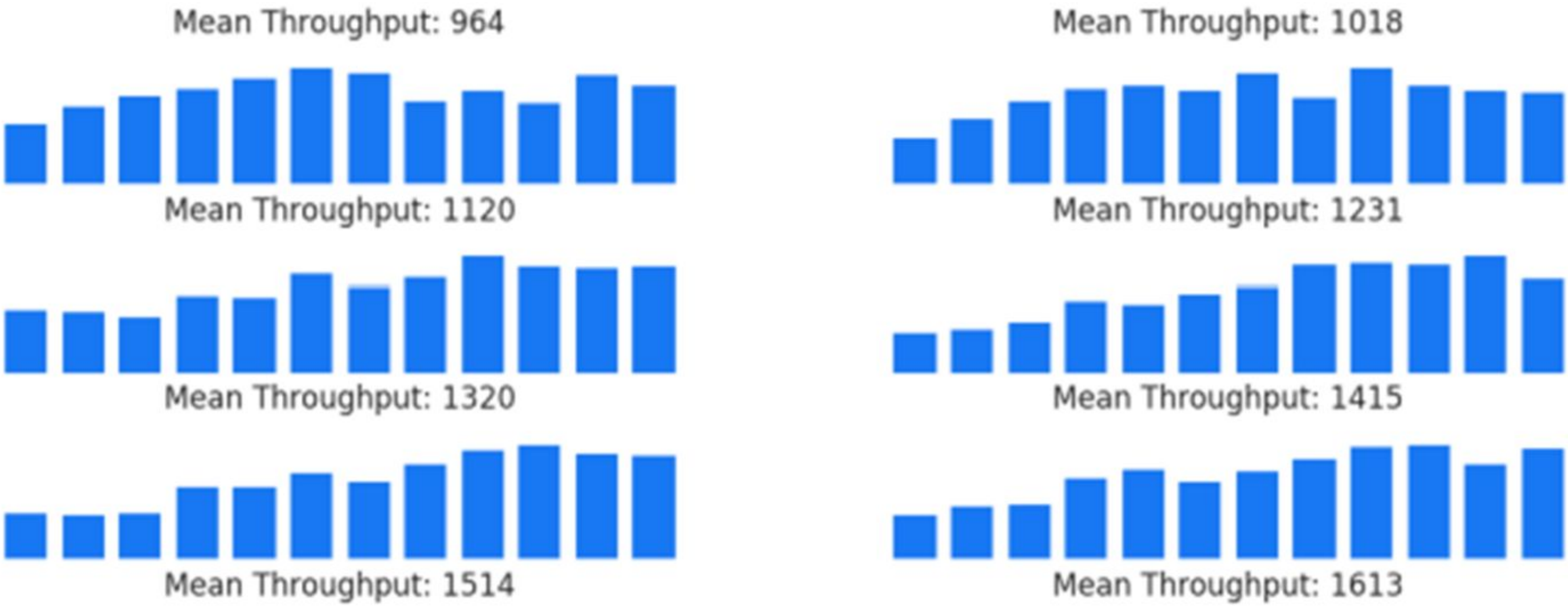
1. Better sampling methods?
2. Unsymmetrical partition?
3. Cyclic window?

Test with kmeans - it’s not about clustering but merging

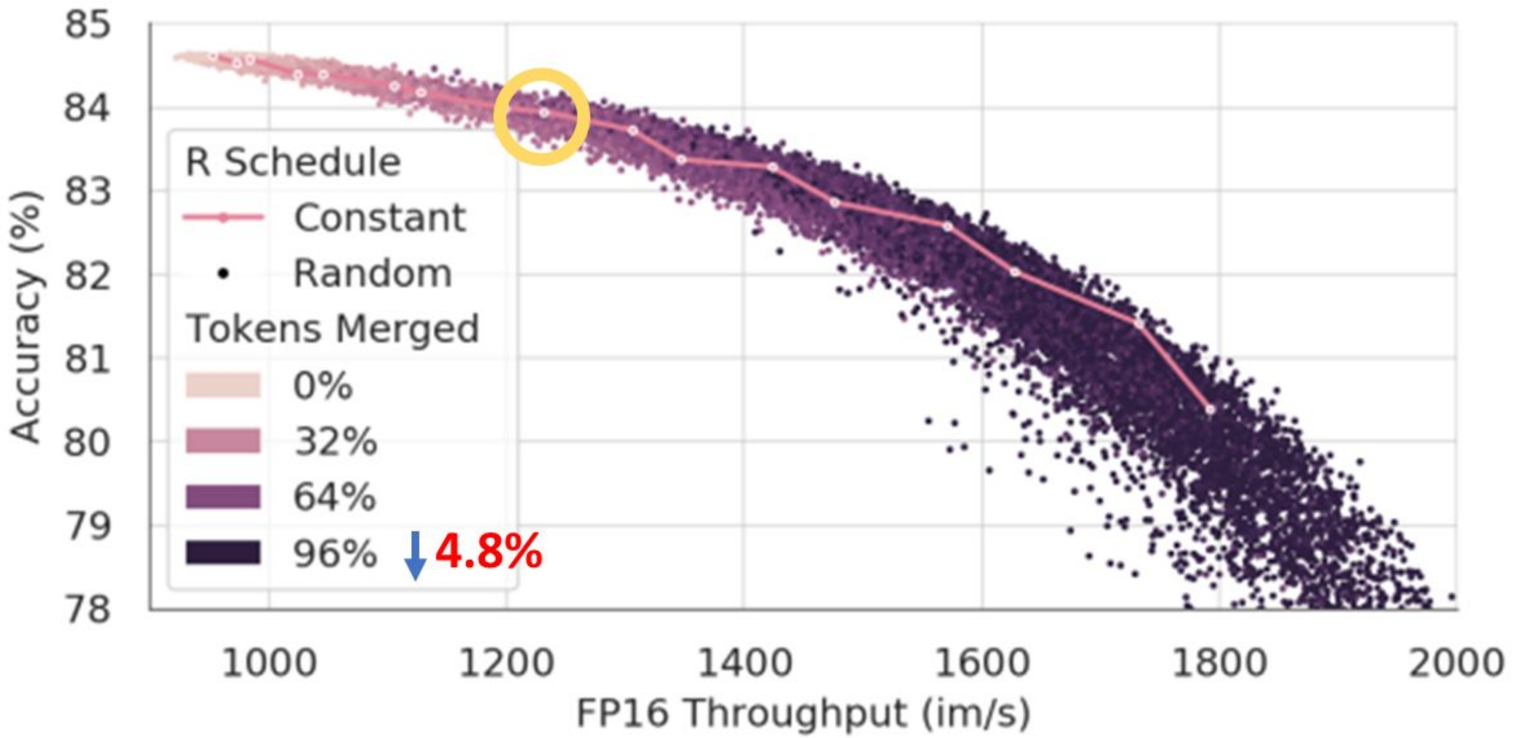
style	algorithm	acc	im/s
prune	random	79.22	**184.4**
prune	attn-based	**79.48**	183.8
merge	kmeans (2 iter)	80.19	169.7
merge	kmeans (5 iter)	80.29	147.5
merge	greedy matching	**84.36**	179.4
merge	bipartite matching	84.25	**182.9**


WHY

Merging schedule



Merging schedule



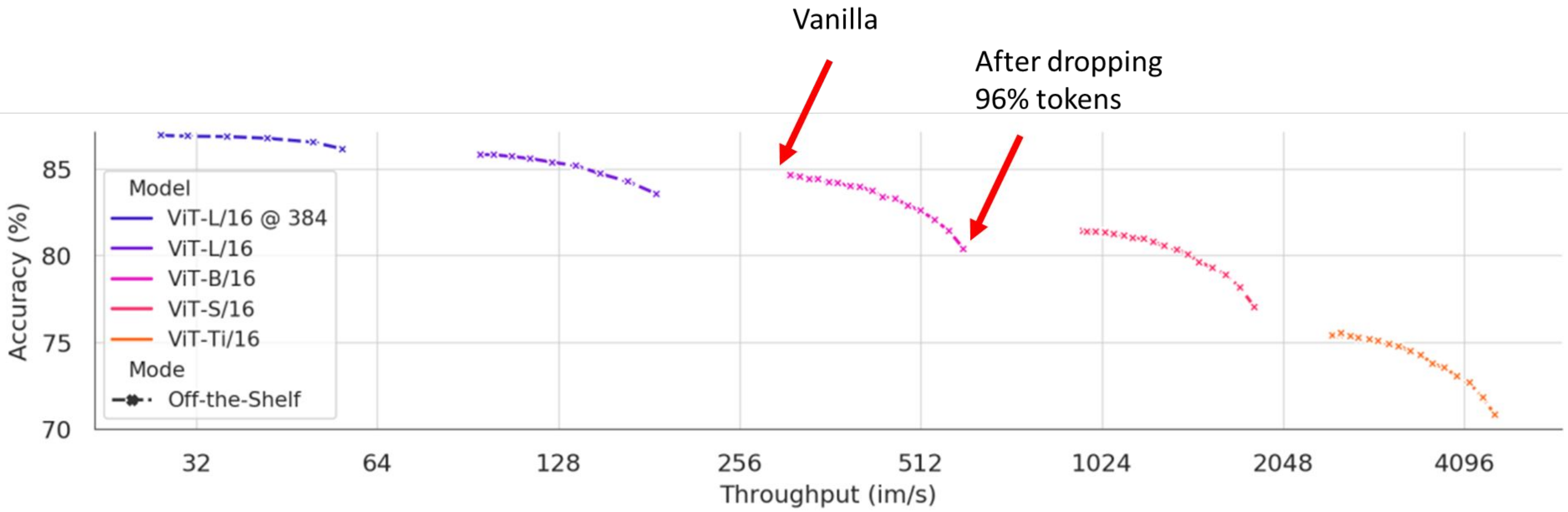
Test with various models

Test settings

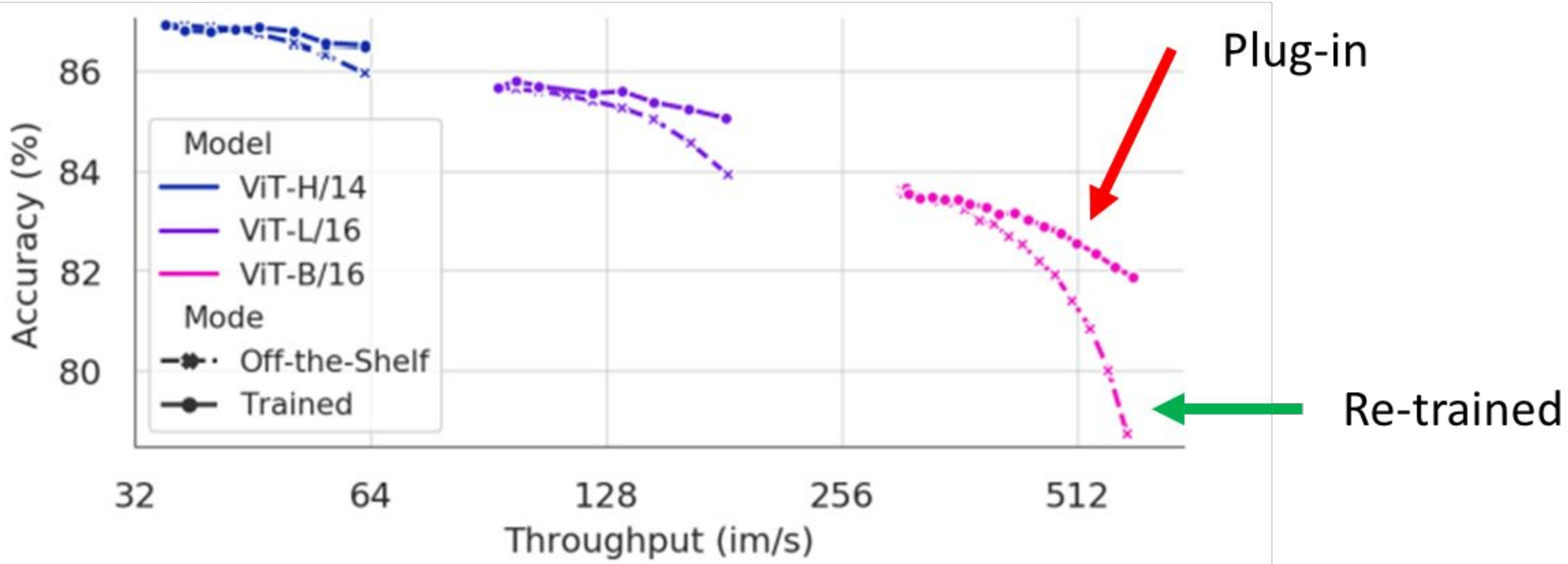
1. Off-the-shelf
   a. Supervised: AugReg, SWAG
   b. Self-supervised: MAE

2. Incorporate the proposed module and retrain a new model with the sampe recipe.

The larger a model is, the less it is affected.



(a) **AugReg Models.** A collection of ImageNet-21k pretrained models (Steiner et al., 2022).



(c) **MAE Models.** Self-supervised models pretrained on ImageNet-1k (He et al., 2022).

“Upgrade with no cost”

	model	input	acc	gflops	im/s
vanilla	ViT-L $^{\text{MAE}}$	224	$85.7^{\dagger}$	61.6	93
	Eff-B6	528	84.0	19.0	$96^{\ddagger}$
	MViTv2-L	224	85.3	42.1	81
	ToMe				
	ViT-H $^{\text{MAE}}_{r7\searrow}$	224	86.1	72.6	81
	ViT-H $^{\text{MAE}}_{r7\rightarrow}$	224	**86.5**	92.9	63
vanilla	ViT-H $^{\text{MAE}}$	224	$86.9^{\dagger}$	167.4	35
	SwinV2-H*	224	85.7	118.1	49

Compared to other pruning methods:

method	acc	inference gflops	inference im/s	train speed
DeiT-S	79.8	4.6	930	1×
A-ViT	78.6†	2.9	-	1×
DynamicViT	79.3	2.9	1505	1×
SP-ViT	79.3	2.6	-	1×
ToMe $^{\text{DeiT}}_{r_{13}\rightarrow}$	**79.4**	2.7	**1552**	**1.5**×
ToMe $^{\text{AugReg}}_{r_{13}\rightarrow}$	79.3	2.7	**1550**	-

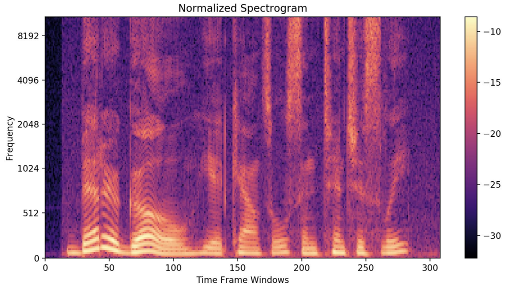
How about other modalities?

Video results

model	input	acc	gflops
ViT-L $^{\text{MAE}}$	16×224^2	84.7	$598\times1\times10$
Swin-L†	32×224^2	83.1	$604\times3\times4$
MViTv2-L	16×224^2	84.3	$377\times1\times10$
ToMe			
ViT-L $^{\text{MAE}}_{r_{55}\rightarrow}$	16×224^2	**84.5**	$325\times1\times4$
ViT-L $^{\text{MAE}}_{r_{65}\rightarrow}$	16×224^2	**84.5**	$\mathbf{281}\times1\times10$
ViT-L $^{\text{MAE}}_{r_{65}\rightarrow}$	16×224^2	84.4	$\mathbf{281}\times1\times10$

Audio

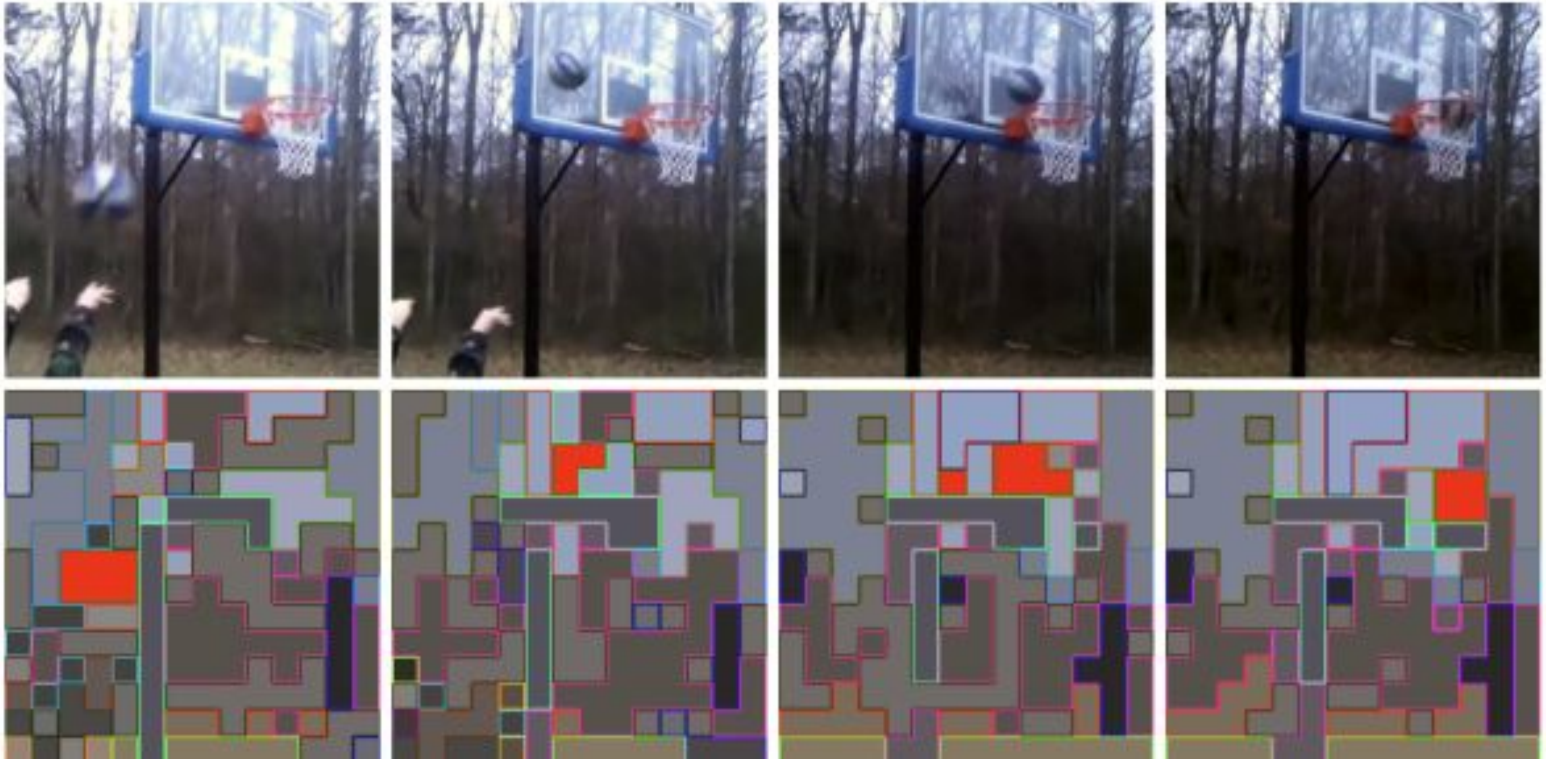
model	mAP	gflops	sample/s
ViT-B$^{\text{MAE}}$	47.3	48.6	103
ViT-B$^{\text{MAE}}_{r_{20}\rightarrow}$	46.2	36.3	136
ViT-B$^{\text{MAE}}_{r_{40}\rightarrow}$	43.1	24.7	**200**
ViT-B$^{\text{MAE}}$	46.4*	48.6	103
ViT-B$^{\text{MAE}}_{r_{20}\rightarrow}$	46.3	36.3	136
ViT-B$^{\text{MAE}}_{r_{40}\rightarrow}$	46.0	24.7	**200**



Visualization

2D attention

3D attention



Thanks

Ablate proportional attention - not solid

src	prop	acc	im/s
mae		**84.25**	**182.9**
mae	✓	83.84	180.9
augreg		82.15	**182.8**
augreg	✓	**83.51**	180.8

(f) **Proportional Attn.** Without MAE pretraining, off-the-shelf models require prop attn.