Paper Review

What Makes for Good Tokenizers in Vision Transformer?

Shengju Qian, et al.

The Chinese University of Hong Kong & Amazon AI

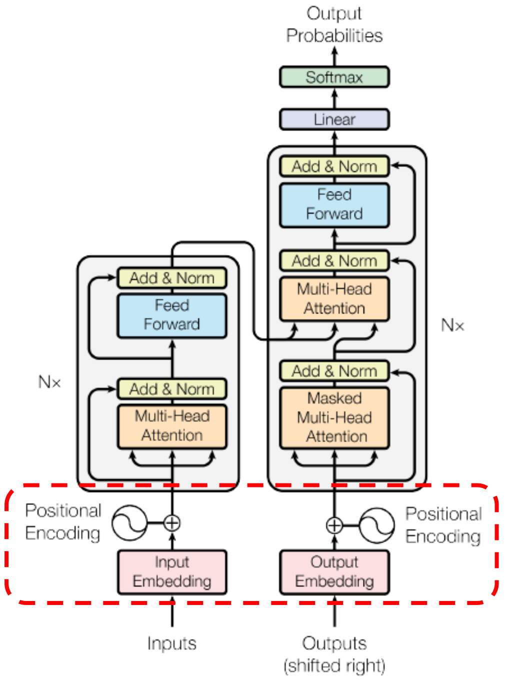
TPAMI 2022

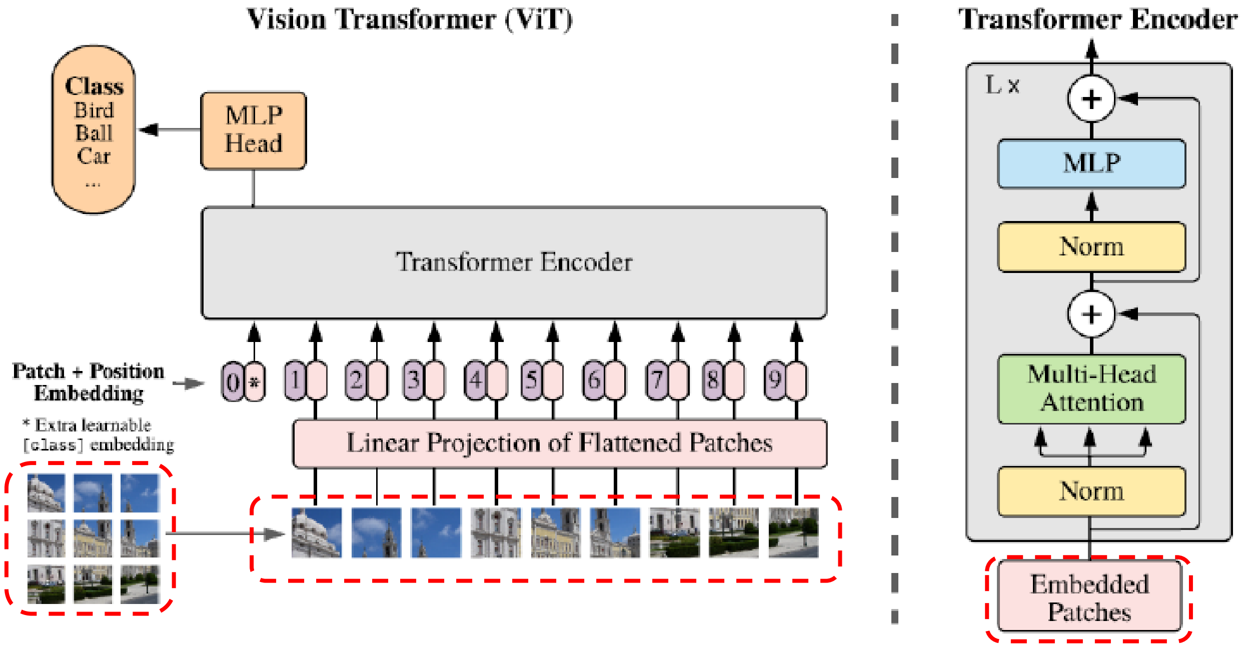
Industrial AI Research, POSCO DX

Vision Lab Susang Kim

Contents

1.Introduction - Vision Transformer





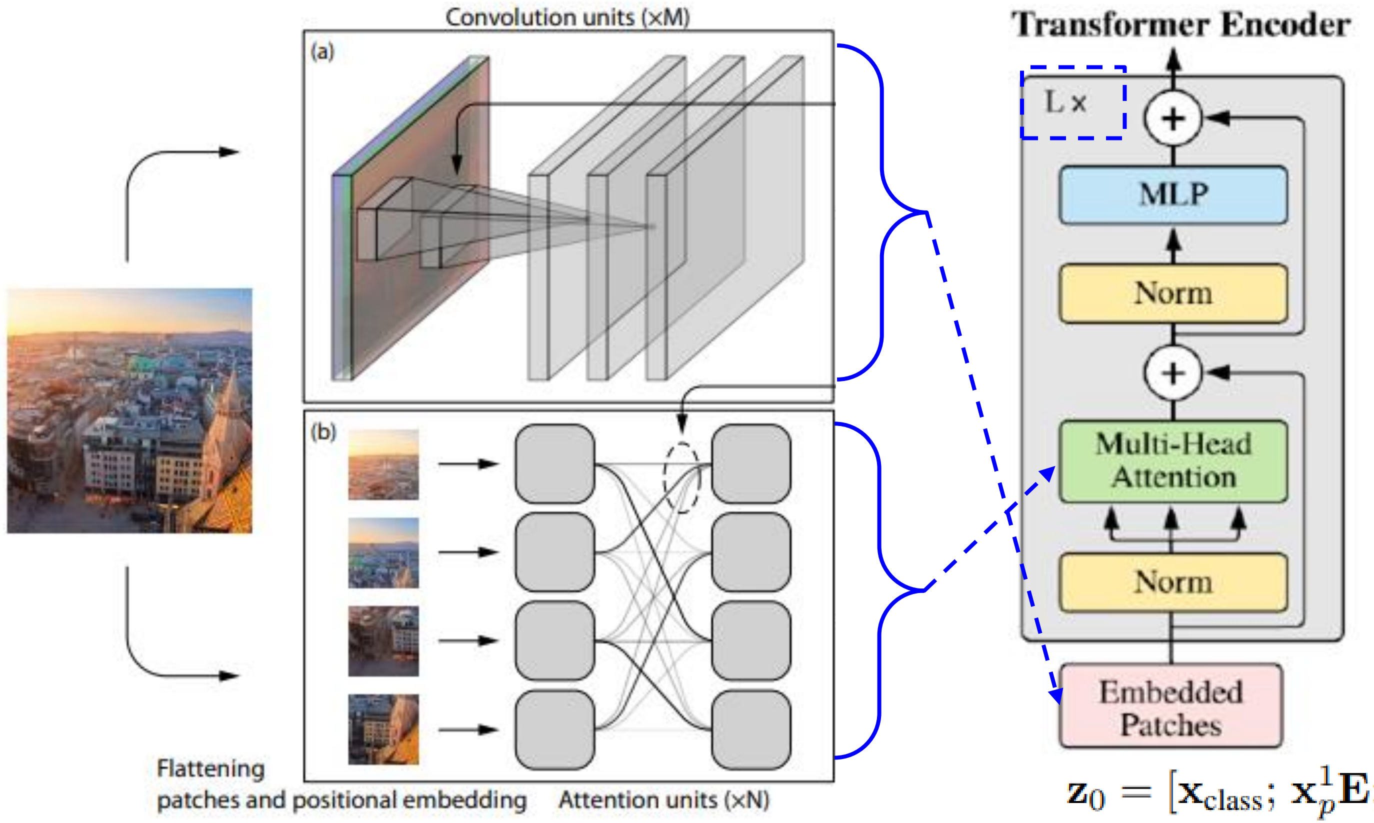
Attention is all you need.

While being **the stemming building block** of transformers, what makes for a good tokenizer has not been well understood in computer vision.
The influence from different tokenizer designs still lacks principled discussion and analysis.

Alexey, et al. An image is worth 16x16 words: Transformers for image recognition at scale. ICLR 2021.

1.Introduction - Convolution vs Self-Attention (Module)

It is difficult for ConvNets to capture long-term dependencies, while self-attention layers are global.



$$\mathbf{x} \in \mathbb{R}^{H \times W \times C}$$

$$N = HW/P^2$$

$$\mathbf{x}_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$$

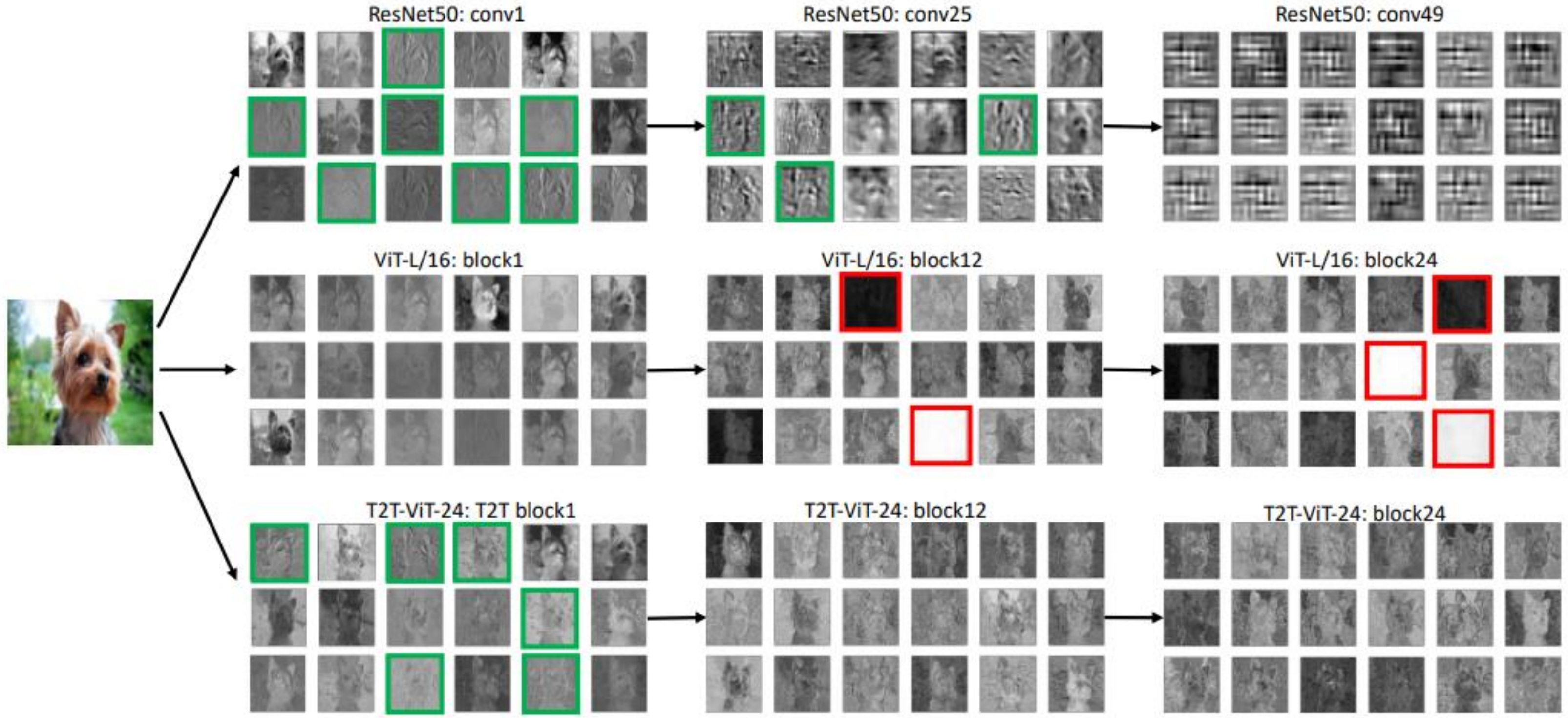
$$\mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}$$

$$\mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D}$$

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\, \cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos},$$

S Tuli et al., “Are Convolutional Neural Networks or Transformers more like human vision?,” CogSci, 2021.

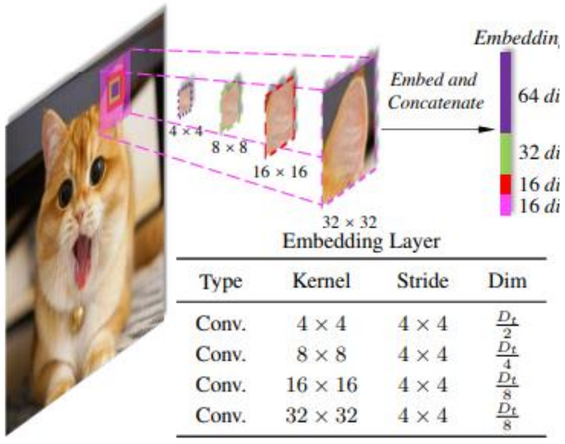
1.Introduction - Feature visualization of local structure



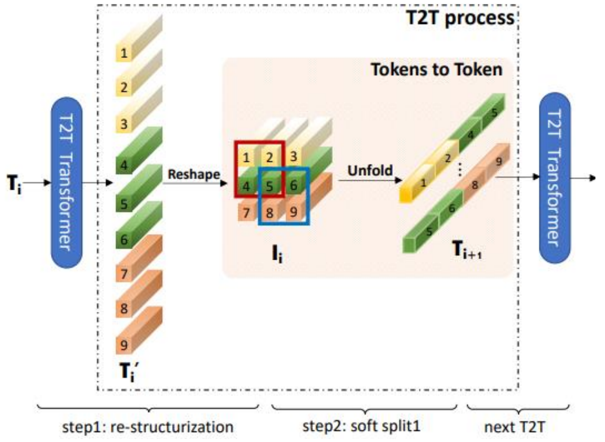
Green boxes highlight learned low-level structure features such as edges and lines;
red boxes highlight invalid feature maps with zero or too large values in ViT.

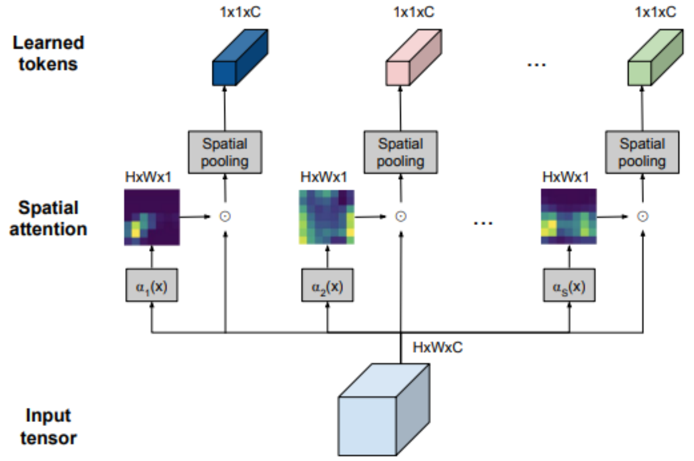
YUAN, Li, et al. Tokens-to-token vit: Training vision transformers from scratch on imagenet. ICCV 2021.

1.Introduction - “patchifying” in Vision Transformer



Type	Kernel	Stride	Dim
Conv.	4 × 4	4 × 4	$\frac{D_t}{2}$
Conv.	8 × 8	4 × 4	$\frac{D_t}{4}$
Conv.	16 × 16	4 × 4	$\frac{D_t}{8}$
Conv.	32 × 32	4 × 4	$\frac{D_t}{8}$

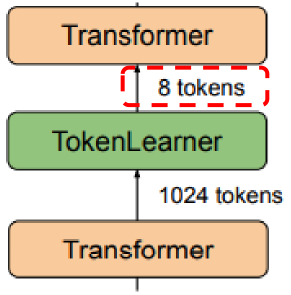




The input image is sampled by four **different kernels** with same stride 4 × 4 in CEL layer.

The tokens Ti are restructurized as an image Ii after **transformation and reshaping**

TokenLearner **learns to spatially attend over a subset of tensor pixels** (i.e., from intermediate spatial representations), and generates a set of token vectors adaptive to the input.



WANG, W., et al. CrossFormer: A versatile vision transformer hinging on cross-scale attention. arXiv 2021.
YUAN, Li, et al. Tokens-to-token vit: Training vision transformers from scratch on imagenet. ICCV 2021.
Michael S., et al. Tokenlearner: What can 8 learned tokens do for images and videos?. NeurIPS 2021.

1.Introduction - Mutual Information

Mutual information initially measures dependencies between random variables. Given A and B, I(A; B) estimates the **"amount of information"** learned from B about the other variable A and vice versa.

$$\mathbf{A} \in \mathcal{R}^{h \times w \times 3}$$
$$\mathbf{B} \in \mathcal{R}^{n_{\text{token}} \times l_{\text{token}}}$$

$$H = -\sum_{i=1}^{N} p_i \log(p_i)$$

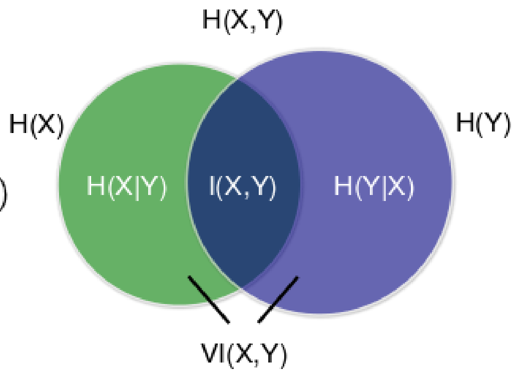
$$I(\mathbf{A};\mathbf{B}) = H(\mathbf{A}) - H(\mathbf{A}|\mathbf{B}) = H(\mathbf{B}) - H(\mathbf{B}|\mathbf{A})$$

$$\begin{aligned} I(\mathbf{A};\mathbf{B}) &= H(\mathbf{A}) - H(\mathbf{A}|\mathbf{B}) \\ &\geq H(\mathbf{A}) - \mathbb{R}(\mathbf{A}|\mathbf{B}) \end{aligned}$$

H(A) = A's marginal entropy
H(A,B) = A,B joint entropy
H(A|B) = A's conditional entropy
R(A|B) = expected error for reconstructing

$$\mathrm{I}(X;Y) = \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p_{(X,Y)}(x,y)\log\frac{p_{(X,Y)}(x,y)}{p_X(x)p_Y(y)} = D_{KL}(P(X,Y)||P(X)P(Y))$$

$$\begin{aligned}
&= \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p_{(X,Y)}(x,y)\log\frac{p_{(X,Y)}(x,y)}{p_X(x)} - \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p_{(X,Y)}(x,y)\log p_Y(y) \\
&= \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p_X(x)p_{Y|X=x}(y)\log p_{Y|X=x}(y) - \sum_{x\in\mathcal{X},y\in\mathcal{Y}} p_{(X,Y)}(x,y)\log p_Y(y) \\
&= \sum_{x\in\mathcal{X}} p_X(x)\left(\sum_{y\in\mathcal{Y}} p_{Y|X=x}(y)\log p_{Y|X=x}(y)\right) - \sum_{y\in\mathcal{Y}}\left(\sum_{x\in\mathcal{X}} p_{(X,Y)}(x,y)\right)\log p_Y(y) \\
&= -\sum_{x\in\mathcal{X}} p_X(x)\mathrm{H}(Y \mid X=x) - \sum_{y\in\mathcal{Y}} p_Y(y)\log p_Y(y) \\
&= -\mathrm{H}(Y \mid X) + \mathrm{H}(Y) \\
&= \mathrm{H}(Y) - \mathrm{H}(Y \mid X).
\end{aligned}$$



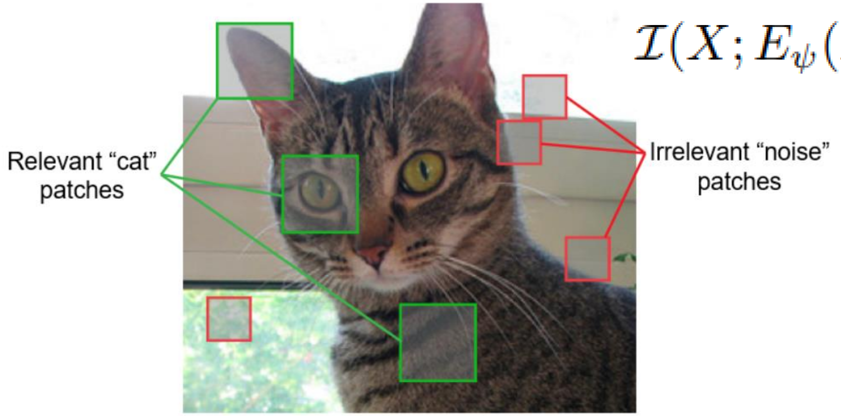
MI = Independent = 0 = entropy

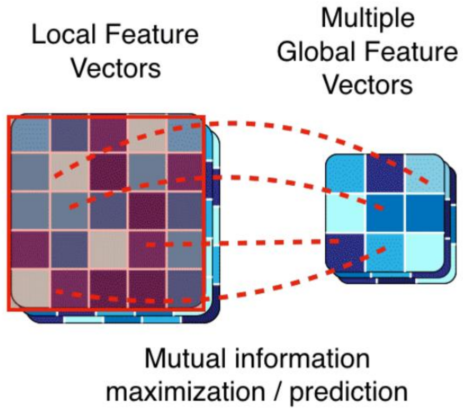
https://en.wikipedia.org/wiki/Mutual_information
https://www.microsoft.com/en-us/research/blog/deep-infomax-learning-good-representations-through-mutual-information-maximization/

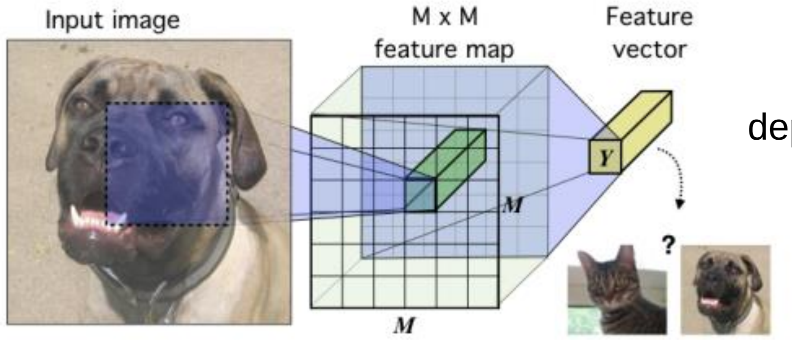
1.Introduction - Mutual Information Maximization

Deep InfoMax (DIM) simultaneously **estimates and maximizes** the mutual information between **input data and learned high-level representations.**

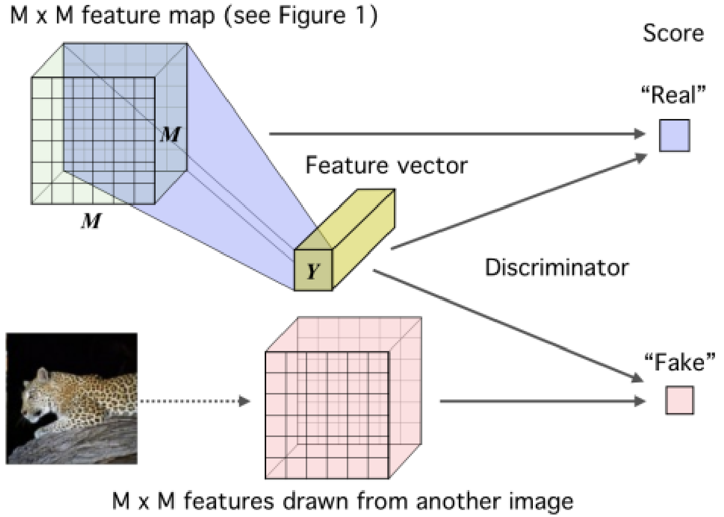
$\mathcal{I}(X; E_\psi(X))$, is maximized.




Local Feature Vectors
Multiple Global Feature Vectors
Mutual information maximization / prediction

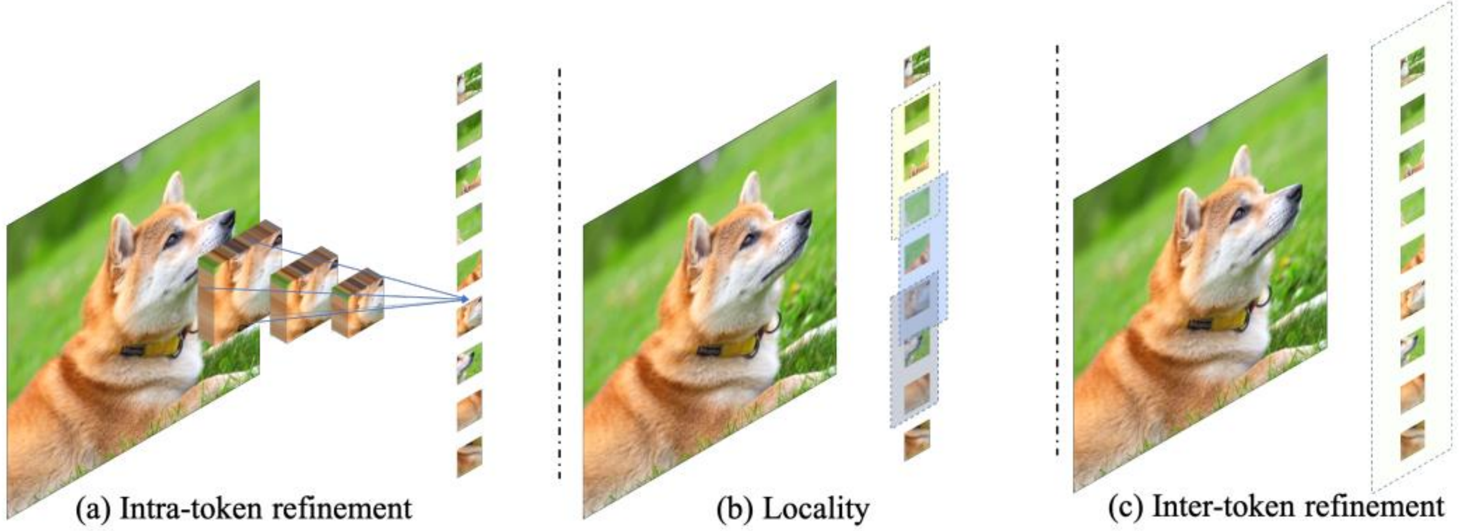


dependent


M x M feature map (see Figure 1)
Score
"Real"
M
M
Feature vector
Y
Discriminator
"Fake"
M x M features drawn from another image

HJELM, R. Devon, et al. Learning deep representations by mutual information estimation and maximization. ICLR 2019.
https://www.microsoft.com/en-us/research/blog/deep-infomax-learning-good-representations-through-mutual-information-maximization/

2.Motivation - Structural Designs for Vision Tokenizer

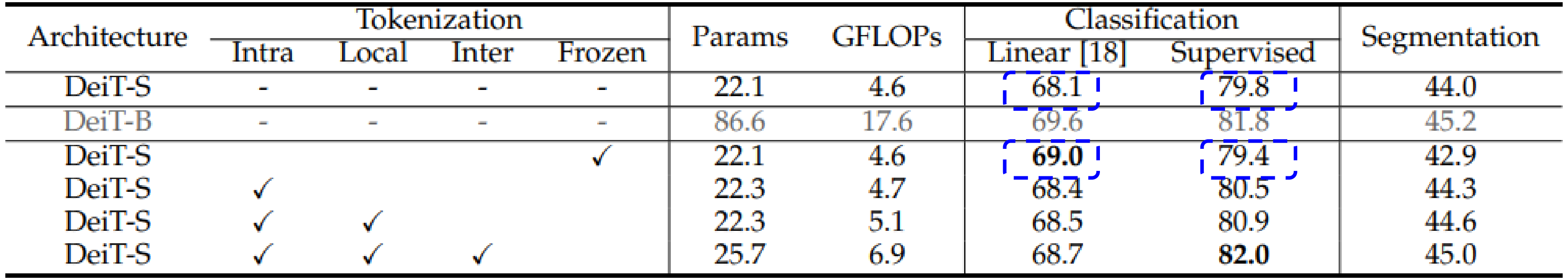


a) Intra-token refinement: stride-p, p × p convolution, naive “patchifying” fails to capture rich context inside the tokens **(multi-scale embeddings, models sub-token)**

b) Locality: Overlapping and uneven token embeddings have been tailored.

c) Inter-token refinement: Modeling intertoken relationship intuitively helps tokenizers encode better features. This refinement suggests performing **global context modeling inside the tokenizers**.

2.Motivation - Performance with different tokenizers



Architecture	Tokenization				Params	GFLOPs	Classification		Segmentation
	Intra	Local	Inter	Frozen			Linear [18]	Supervised	
DeiT-S	-	-	-	-	22.1	4.6	68.1	79.8	44.0
DeiT-B	-	-	-	-	86.6	17.6	69.6	81.8	45.2
DeiT-S				✓	22.1	4.6	**69.0**	79.4	42.9
DeiT-S	✓				22.3	4.7	68.4	80.5	44.3
DeiT-S	✓	✓			22.3	5.1	68.5	80.9	44.6
DeiT-S	✓	✓	✓		25.7	6.9	68.7	**82.0**	45.0

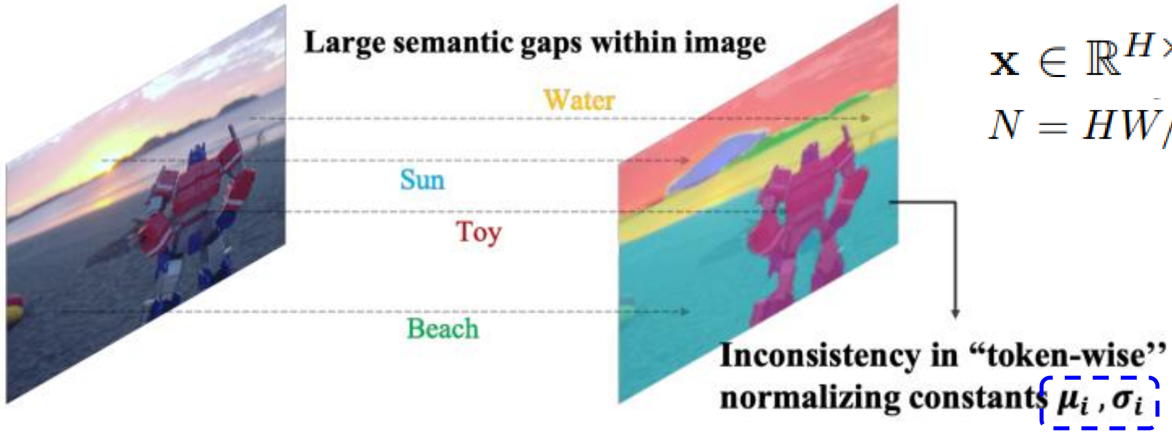
Performance on various vision tasks with different tokenizers. “Intra”, “Local”, and “Inter” respectively refer to applying the intra-token, locality, and inter-token refinement strategies. “Frozen” refers to the frozen randomly-initialized tokenization in MoCov3.

1) Empirical correlations exist between **increasing token diversity and better results. (more checking)**
2) Maximizing **conditional entropy doesn’t guarantee** better performance. H(A|B) = A’s conditional entropy
   - Randomly-initialized frozen tokenizer maximizes conditional entropy and token diversity, it gives inferior performance
3) Tokenizers might **influence optimization** across tasks.(better linear performance with frozen token.)
4) The goal of tokenizer is to maintain a **trade-off** between **feature expressiveness and information accessibility.**

Motivated by these findings, we expect our tokenizer to **simultaneously maintain the information accessibility and feature quality**. Therefore, we further study versatile and unexplored design strategies for vision tokenizers, through normalization and optimization.

2.Motivation - Normalization in Vision Tokenizer

Although the importance of normalization has been acknowledged in transformer for both language and vision, **it has rarely been explored inside the tokenizer.**
Considering the semantic variations between images, patch tokens tend to encounter semantic variation



Large semantic gaps within image
Water
Sun
Toy
Beach
Inconsistency in "token-wise" normalizing constants μ_i , σ_i

$$\mathbf{x} \in \mathbb{R}^{H \times W \times C} \qquad \mathbf{x}_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$$

$$N = HW/P^2$$

The addition of **batch normalization (BN) to the patchify stem worsens its accuracy.**
According to previous analysis normalization layers tend to **"wash away" texture and semantic information**. Such "filtered" effect inevitably reduces token diversity and possibly cuts off necessary diversity in transformer.

Fig. 3. **Example about the large semantic gap across regions and tokens.** This difference leads to distinctive statistics. Using fixed normalizing constants would inevitably "filter" the original information and reduces the token diversity.

$$X_{BN} = \frac{X - \mu_{B,H,W}}{\sigma_{B,H,W}} * \gamma_{BN} + \beta_{BN},$$

3.Method - “wash away” semantic information (SPADE)

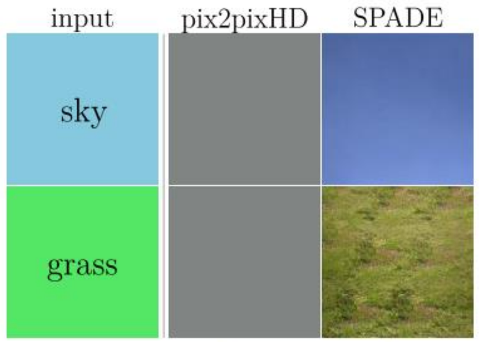
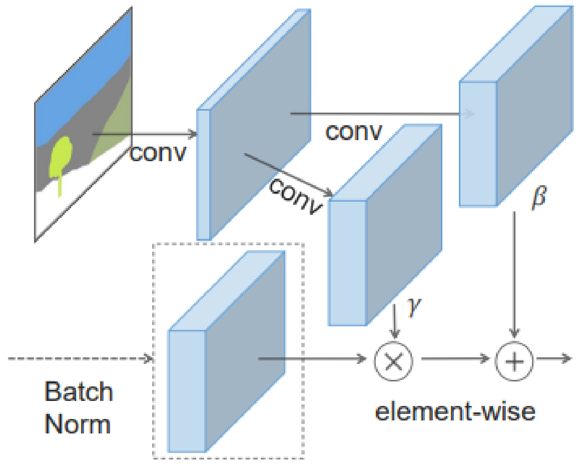
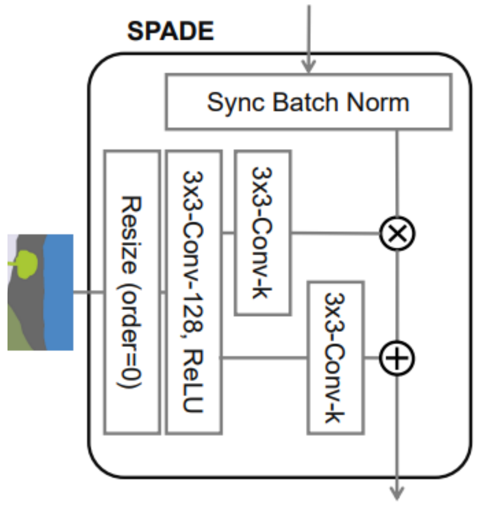


Figure 3: Comparing results given uniform segmentation maps: while the SPADE generator produces plausible textures, the pix2pixHD generator [48] produces two identical outputs due to the loss of the semantic information after the normalization layer.

$$\mu_c^i = \frac{1}{NH^iW^i}\sum_{n,y,x} h^i_{n,c,y,x}$$

$$\sigma_c^i = \sqrt{\frac{1}{NH^iW^i}\sum_{n,y,x}\left((h^i_{n,c,y,x})^2 - (\mu_c^i)^2\right)}.$$





$$\gamma^i_{c,y,x}(\mathbf{m})\frac{h^i_{n,c,y,x} - \mu^i_c}{\sigma^i_c} + \beta^i_{c,y,x}(\mathbf{m})$$

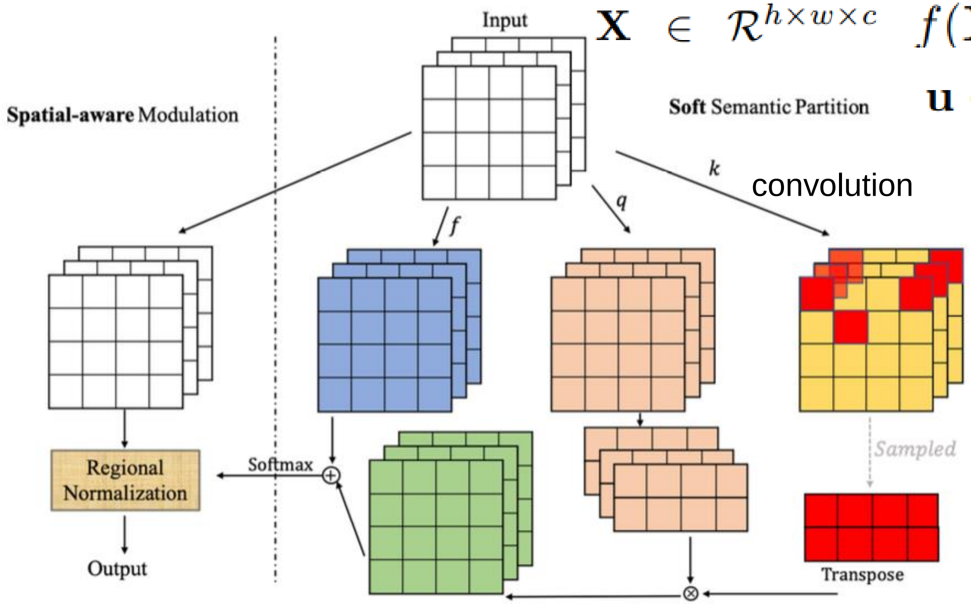
m to the scaling and bias values at the site (c, y, x) in the i-th activation map.

The spatially-adaptive normalization, which utilizes the input semantic layout while performing the affine transformation in the normalization layers.

PARK, Taesung, et al. Semantic image synthesis with spatially-adaptive normalization. CVPR 2019.

3.Method - Modulation Across Tokens (MoTo)

To modulate the diverse semantics in input using regional statistics. This strategy not only formulates **feature distributions**, but also provides the tokenizer with more **plausible semantic content.**



convolution

$\mathbf{X} \in \mathcal{R}^{h\times w\times c}$ $f(\mathbf{X}) \in \mathcal{R}^{h\times w\times n}$ $\mathbf{Z} \in \mathcal{R}^{h\times w\times n}$

$\mathbf{u} \in \mathcal{R}^{1\times 1\times n}$ $\mathbf{L} \in \mathcal{R}^{h\times w\times n}$

soft semantic layout

$$\mathbf{Z} = \mathbf{u}\cdot k_n(\mathbf{X})^T q(\mathbf{X}) + f(\mathbf{X})$$

n feature points (uniformly sampled)

$\mathbf{L}_k \in \mathcal{R}^{h\times w}$ the probability of the spatial pixels belonging to entity k

$$\mathbf{L}_k = \frac{\exp(\tau \mathbf{Z}_k)}{\sum_{i=1}^{n} \exp(\tau \mathbf{Z}_i)}$$

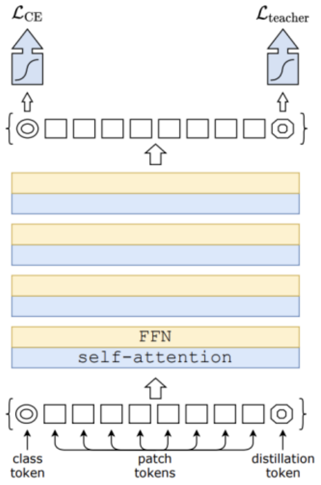
$$\mathrm{MoTo}(\mathbf{X}) = \sum_{i=1}^{n}\left(\frac{\mathbf{X} - \mu(\mathbf{X}\odot \mathbf{L}_i)}{\sigma(\mathbf{X}\odot \mathbf{L}_i) + \epsilon} \times \beta_i + \alpha_i\right) \odot \mathbf{L}_i$$

where k ∈ [0, n − 1] and τ is the temperature coefficient set to 0.1. default n = 8

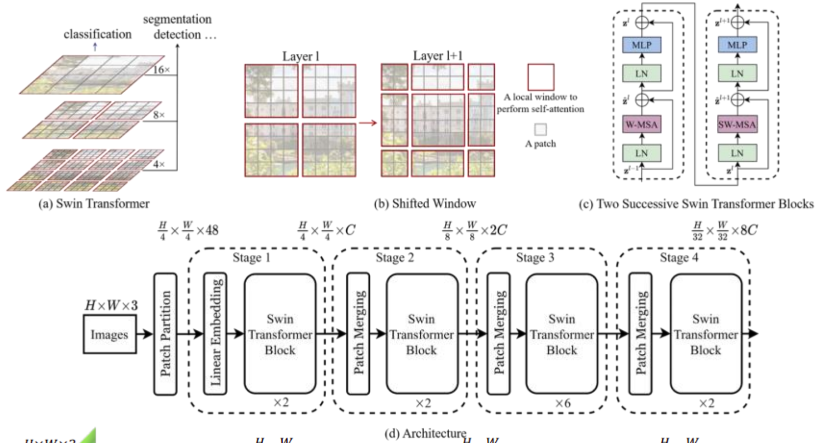
MoTo incorporates global context to vision tokenizer using normalization.

3.Method - On Various Transformer Architectures.

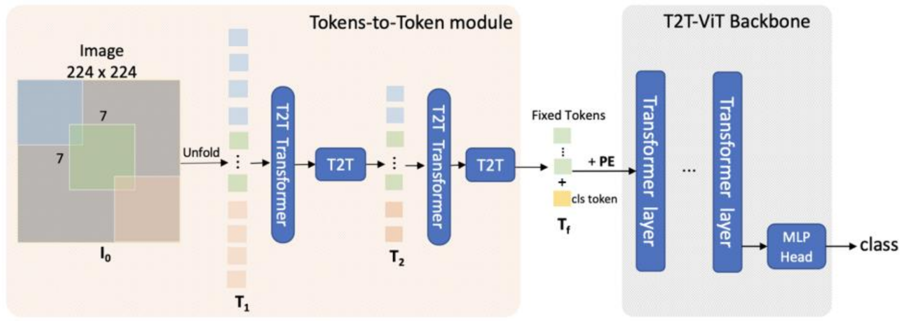
DeiT



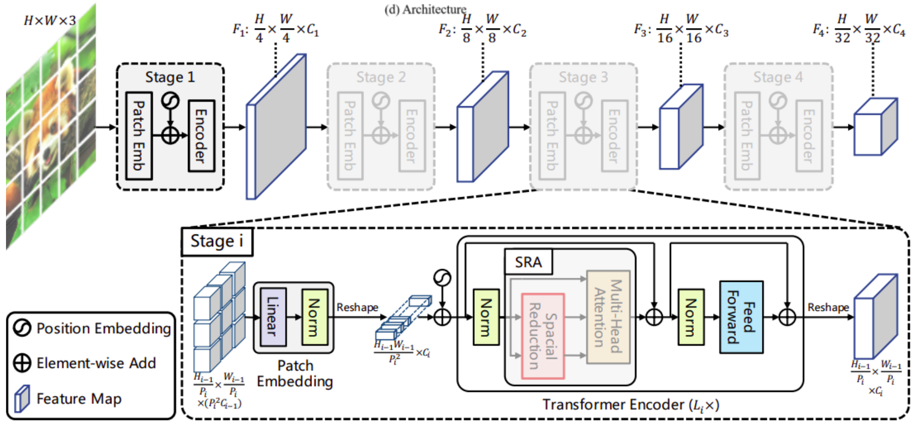
Swin



T2T-ViT



PVT

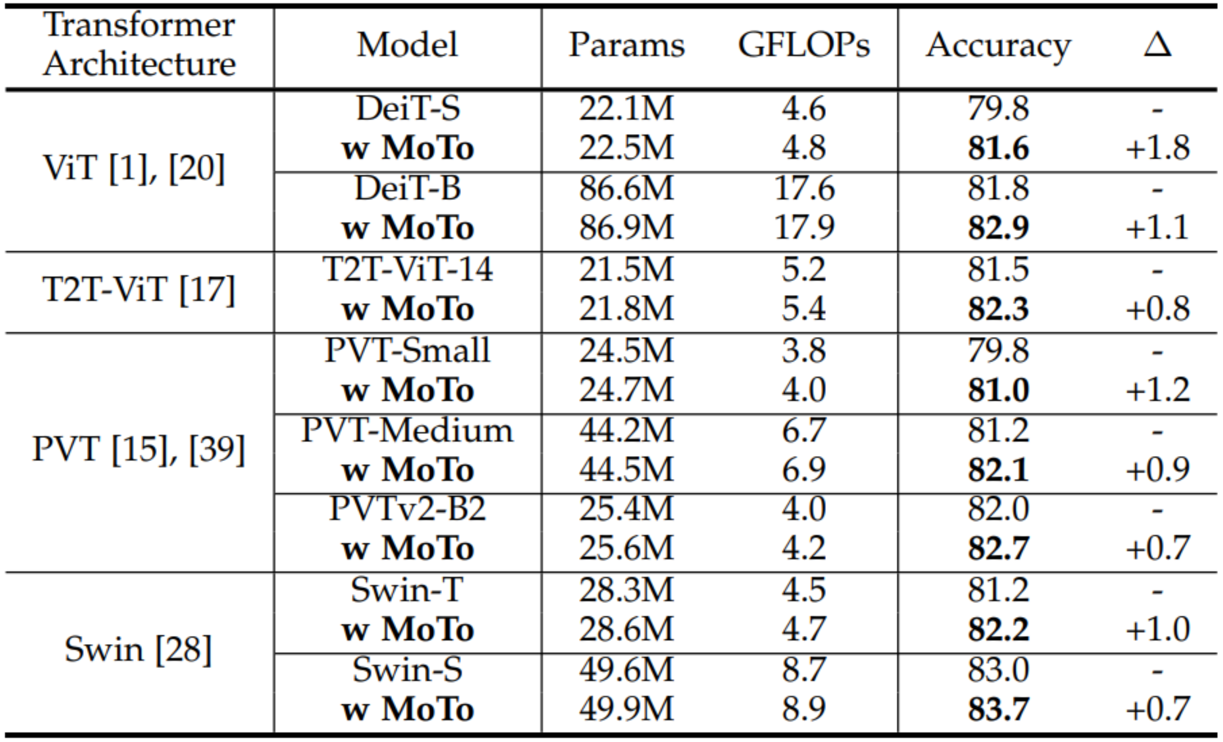


$H\times W\times3$
$F_1: \frac{H}{4}\times\frac{W}{4}\times C_1$
$F_2: \frac{H}{8}\times\frac{W}{8}\times C_2$
$F_3: \frac{H}{16}\times\frac{W}{16}\times C_3$
$F_4: \frac{H}{32}\times\frac{W}{32}\times C_4$
Stage 1
Stage 2
Stage 3
Stage 4
Patch Emb
Encoder
Stage i
Linear
Norm
Patch Embedding
Reshape
SRA
Spacial Reduction
Multi-Head Attention
Feed Forward
Transformer Encoder ($L_i\times$)
Position Embedding
Element-wise Add
Feature Map

Ze, et al. Swin transformer: Hierarchical vision transformer using shifted windows. ICCV 2021.
Wenhai, et al. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. ICCV 2021.
Hugo, et al. Training data-efficient image transformers & distillation through attention. ICML 2021.
Li, et al. Tokens-to-token vit: Training vision transformers from scratch on imagenet. ICCV 2021.

3.Method - Performance of image recognition on ImageNet

To further validate that MoTo is versatile with different tokenizers and transformer architectures, we integrate the proposed module to several state-of-the-art methods



Transformer Architecture	Model	Params	GFLOPs	Accuracy	Δ
ViT [1], [20]	DeiT-S	22.1M	4.6	79.8	-
	w MoTo	22.5M	4.8	**81.6**	+1.8
	DeiT-B	86.6M	17.6	81.8	-
	w MoTo	86.9M	17.9	**82.9**	+1.1
T2T-ViT [17]	T2T-ViT-14	21.5M	5.2	81.5	-
	w MoTo	21.8M	5.4	**82.3**	+0.8
PVT [15], [39]	PVT-Small	24.5M	3.8	79.8	-
	w MoTo	24.7M	4.0	**81.0**	+1.2
	PVT-Medium	44.2M	6.7	81.2	-
	w MoTo	44.5M	6.9	**82.1**	+0.9
	PVTv2-B2	25.4M	4.0	82.0	-
	w MoTo	25.6M	4.2	**82.7**	+0.7
Swin [28]	Swin-T	28.3M	4.5	81.2	-
	w MoTo	28.6M	4.7	**82.2**	+1.0
	Swin-S	49.6M	8.7	83.0	-
	w MoTo	49.9M	8.9	**83.7**	+0.7

the consistent improvements are made across different transformer architectures.

3.Method - Complexity Analysis

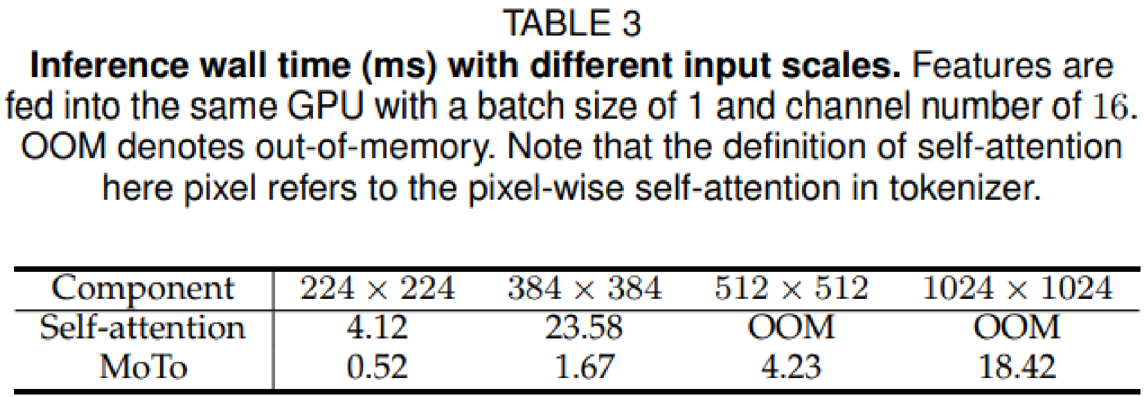


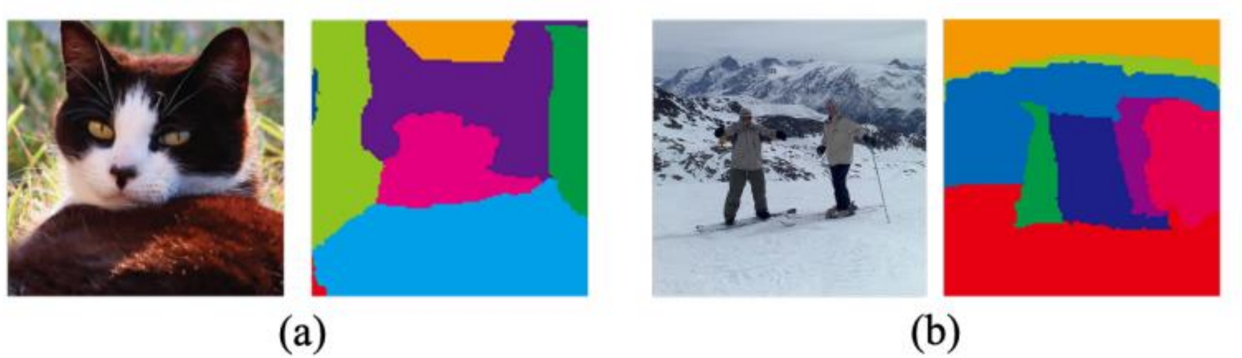
TABLE 3
Inference wall time (ms) with different input scales. Features are fed into the same GPU with a batch size of 1 and channel number of 16. OOM denotes out-of-memory. Note that the definition of self-attention here pixel refers to the pixel-wise self-attention in tokenizer.

Component	224×224	384×384	512×512	1024×1024
Self-attention	4.12	23.58	OOM	OOM
MoTo	0.52	1.67	4.23	18.42

MoTo incorporates **global context modeling to vision tokenizer using normalization.**
MoTo provides a **computational-friendly** choice to perform intertoken modulation. $\mathcal{O}(NCHW)$

The actual computation of injecting **pixel-wise regional information with self-attention scales quadratically**, which is unfeasible for processing high-resolution images. $\mathcal{O}(C(HW)^2)$

3.Method - Visualization of semantic layout



$$\mathbf{L}_{i,j} \in \mathcal{R}^{1\times1\times k}$$

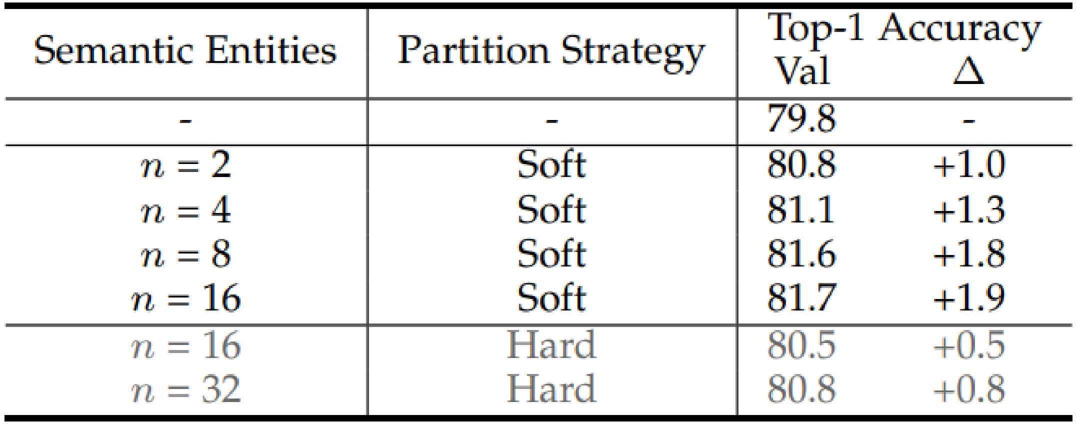
we colorize each location (i, j) using the index with the highest probability.

Visualizations of the self-learned semantic layout. Both image (a) and (b) are testing images. Each color shown in the layout denotes a semantic entity in the soft layout L.

the semantic entities highlight the difference between foreground, background, and instances.
Notably, the entities represent some details on cat face.
Normalizing these feature points with semantic grouping improves the feature quality of tokens.

$$\text{MoTo}(\mathbf{X}) = \sum_{i=1}^{n}\left(\frac{\mathbf{X} - \mu(\mathbf{X} \odot \mathbf{L}_i)}{\sigma(\mathbf{X} \odot \mathbf{L}_i) + \epsilon} \times \beta_i + \alpha_i\right) \odot \mathbf{L}_i$$

3.Method - Ablations about MoTo

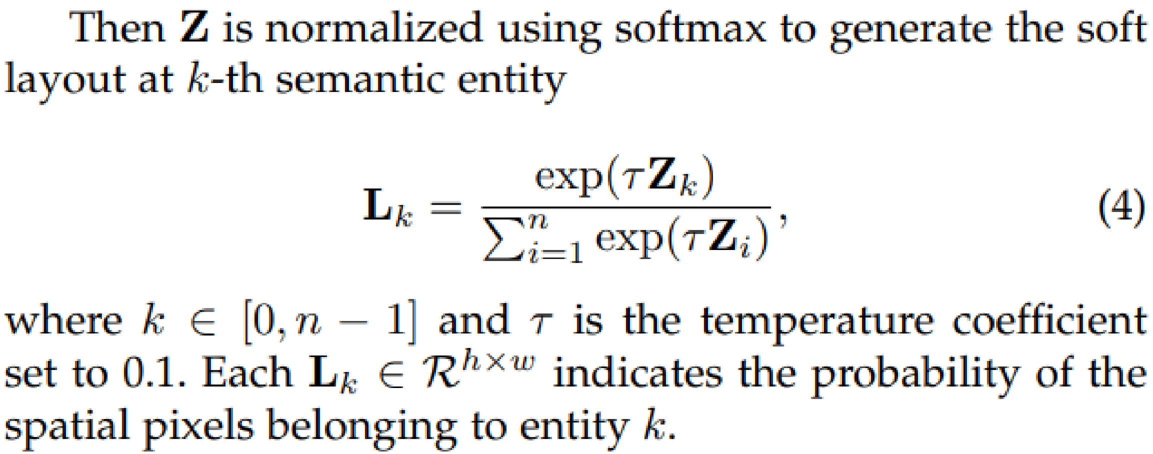


Semantic Entities	Partition Strategy	Top-1 Accuracy	
		Val	Δ
-	-	79.8	-
n = 2	Soft	80.8	+1.0
n = 4	Soft	81.1	+1.3
n = 8	Soft	81.6	+1.8
n = 16	Soft	81.7	+1.9
n = 16	Hard	80.5	+0.5
n = 32	Hard	80.8	+0.8

choose n = 8 for most experiments. considering the tradeoff between effectiveness and complexity.

$\mathbf{L} \in \mathcal{R}^{h\times w\times n}$

soft semantic layout



Then $\mathbf{Z}$ is normalized using softmax to generate the soft layout at k-th semantic entity

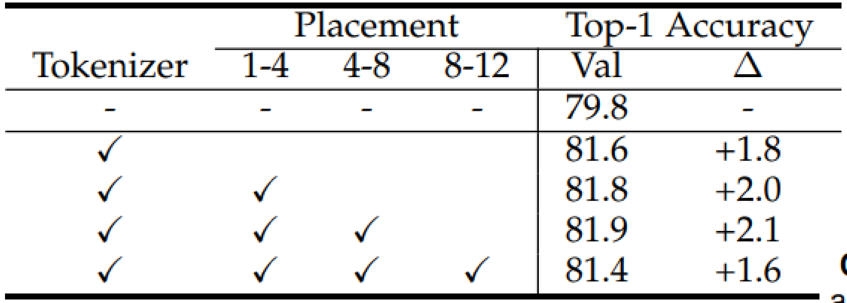
$$\mathbf{L}_k = \frac{\exp(\tau \mathbf{Z}_k)}{\sum_{i=1}^{n} \exp(\tau \mathbf{Z}_i)}, \quad (4)$$

where $k \in [0, n-1]$ and τ is the temperature coefficient set to 0.1. Each $\mathbf{L}_k \in \mathcal{R}^{h\times w}$ indicates the probability of the spatial pixels belonging to entity k.

we implement **argmax** on the activation maps spatially and further obtain a class map to perform **hard partition**.

3.Method - Experimental Analysis with MoTo

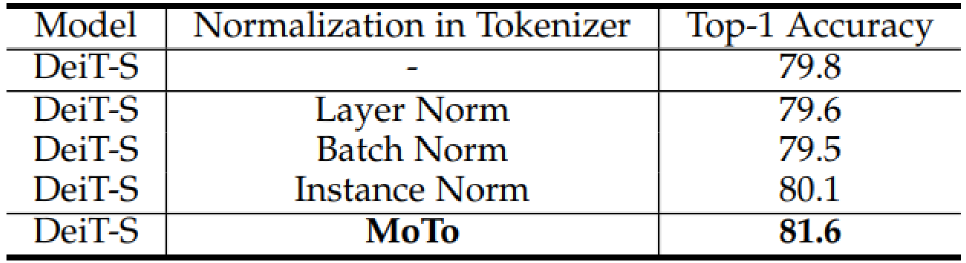
TABLE 5
Ablation study on absorbing MoTo into transformer blocks. The baseline architecture uses DeiT-S. We adopt a MoTo layer with 8 semantic entities and ensemble it into transformer blocks. The placement denotes the layer number of transformer block that adopts MoTo.



	Placement			Top-1 Accuracy	
Tokenizer	1-4	4-8	8-12	Val	Δ
-	-	-	-	79.8	-
✓				81.6	+1.8
✓	✓			81.8	+2.0
✓	✓	✓		81.9	+2.1
✓	✓	✓	✓	81.4	+1.6

Absorbing MoTo into transformer blocks
We further conduct a pilot study to ensemble MoTo at **the end of each transformer layer**.

TABLE 6
Comparison of different normalizations in tokenizer. The baseline architecture uses DeiT-S. Each row denotes result trained with different normalization strategy. Top-1 Accuracy denotes the validation accuracy on ImageNet.



Model	Normalization in Tokenizer	Top-1 Accuracy
DeiT-S	-	79.8
DeiT-S	Layer Norm	79.6
DeiT-S	Batch Norm	79.5
DeiT-S	Instance Norm	80.1
DeiT-S	**MoTo**	**81.6**

Comparison with different normalizations.
As shown in Table 6, **both BN and LN slightly harm the model's performance.** As analyzed, such normalization methods bring "filtered" effect on tokens' features.

3.Method - TokenProp Objective

The core idea of TokenProp is to provide optimization objectives for vision tokenizer, which lead to **better token representations** by maintaining the trade-off between feature fineness and information accessibility.

$$I(\mathbf{A};\mathbf{B}) = H(\mathbf{A}) - H(\mathbf{A}|\mathbf{B}) \geq H(\mathbf{A}) - \mathbb{R}(\mathbf{A}|\mathbf{B})$$

$$\arg\max I(\mathbf{A};\mathbf{B}) = \arg\max -\mathbb{R}(\mathbf{A}|\mathbf{B})$$

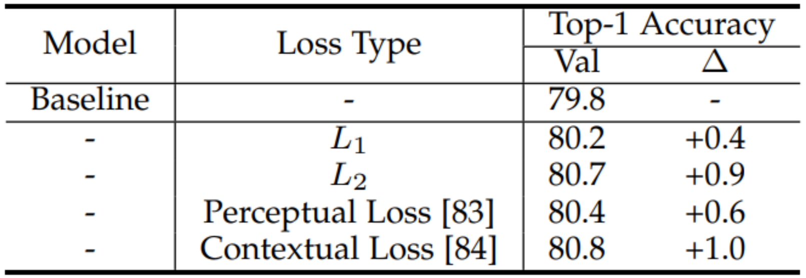
dependent

$$= \arg\max \mathbb{E}_{q(\mathbf{A},\mathbf{B})}[\log q(\mathbf{A}|\mathbf{B})]$$

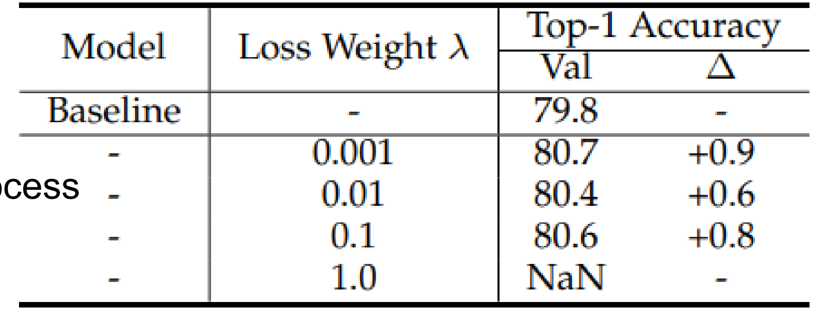
we estimate the parameters of q(A|B) using a decoder network G_θ. Ideally, zero information loss through the tokenization process retains all information including the useful one.

$$\underset{\phi,\omega,\theta}{\text{minimize}} \mathcal{L}(F_\omega(F_\phi(\mathbf{x}));\mathbf{y}) + \lambda\mathcal{L}_{rec}(G_\theta(F_\phi(\mathbf{x}));\mathbf{x})$$

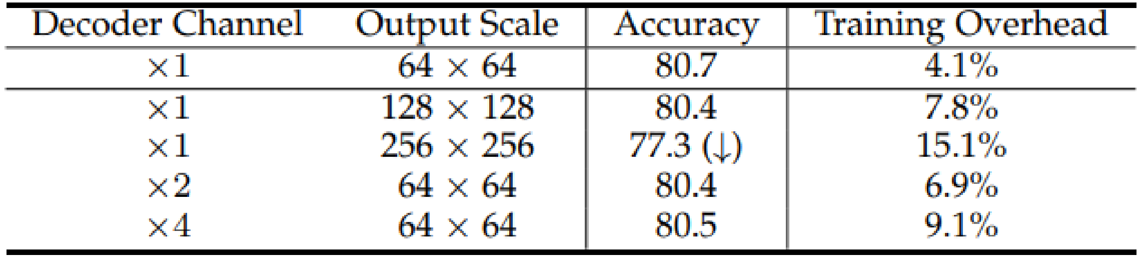
x : input, y : label, φ, ω ,θ : parameters of respective modules



Model	Loss Type	Top-1 Accuracy	
		Val	Δ
Baseline	-	79.8	-
-	L_1	80.2	+0.4
-	L_2	80.7	+0.9
-	Perceptual Loss [83]	80.4	+0.6
-	Contextual Loss [84]	80.8	+1.0



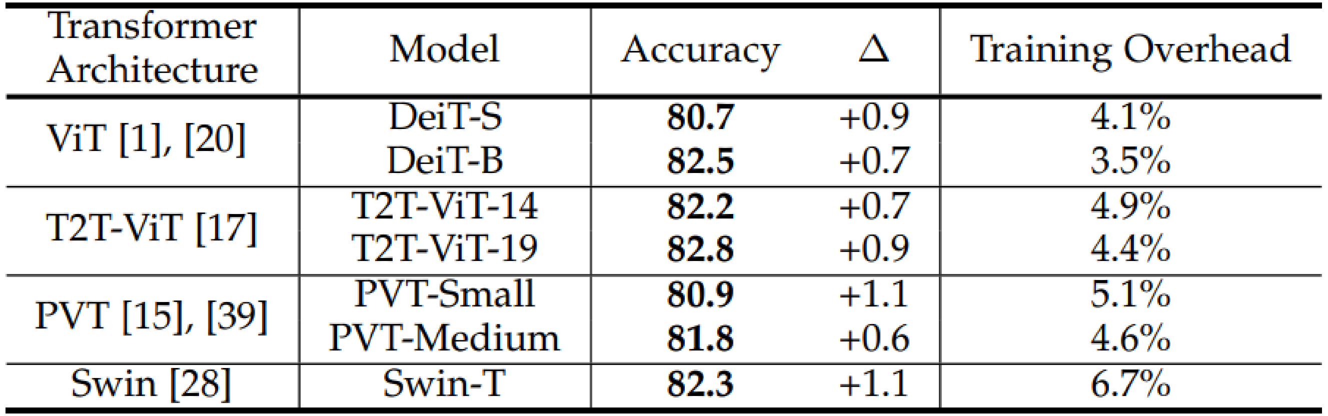
Model	Loss Weight λ	Top-1 Accuracy	
		Val	Δ
Baseline	-	79.8	-
-	0.001	80.7	+0.9
-	0.01	80.4	+0.6
-	0.1	80.6	+0.8
-	1.0	NaN	-



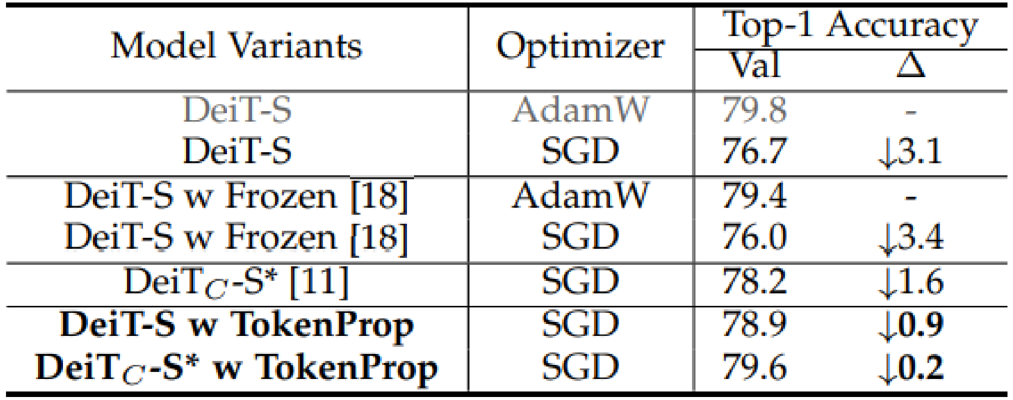
Decoder Channel	Output Scale	Accuracy	Training Overhead
×1	64 × 64	80.7	4.1%
×1	128 × 128	80.4	7.8%
×1	256 × 256	77.3 (↓)	15.1%
×2	64 × 64	80.4	6.9%
×4	64 × 64	80.5	9.1%

4.Experiments - ImageNet validation with TokenProp

TokenProp serves as a regularization term that
provides the encoded features from tokenizers with more diversity.



Transformer Architecture	Model	Accuracy	Δ	Training Overhead
ViT [1], [20]	DeiT-S	**80.7**	+0.9	4.1%
	DeiT-B	**82.5**	+0.7	3.5%
T2T-ViT [17]	T2T-ViT-14	**82.2**	+0.7	4.9%
	T2T-ViT-19	**82.8**	+0.9	4.4%
PVT [15], [39]	PVT-Small	**80.9**	+1.1	5.1%
	PVT-Medium	**81.8**	+0.6	4.6%
Swin [28]	Swin-T	**82.3**	+1.1	6.7%



Model Variants	Optimizer	Top-1 Accuracy	
		Val	Δ
DeiT-S	AdamW	79.8	-
DeiT-S	SGD	76.7	↓3.1
DeiT-S w Frozen [18]	AdamW	79.4	-
DeiT-S w Frozen [18]	SGD	76.0	↓3.4
DeiT$_C$-S* [11]	SGD	78.2	↓1.6
DeiT-S w TokenProp	SGD	78.9	**↓0.9**
DeiT$_C$-S* w TokenProp	SGD	79.6	**↓0.2**

TokenProp enables the adaptability of the model towards SGD, which significantly reduces the performance drop.

4.Experiments - MoTo and TokenProp / Data inefficiency

We can see that the models could be further boosted, suggesting the compatibility of structural and objective designs.

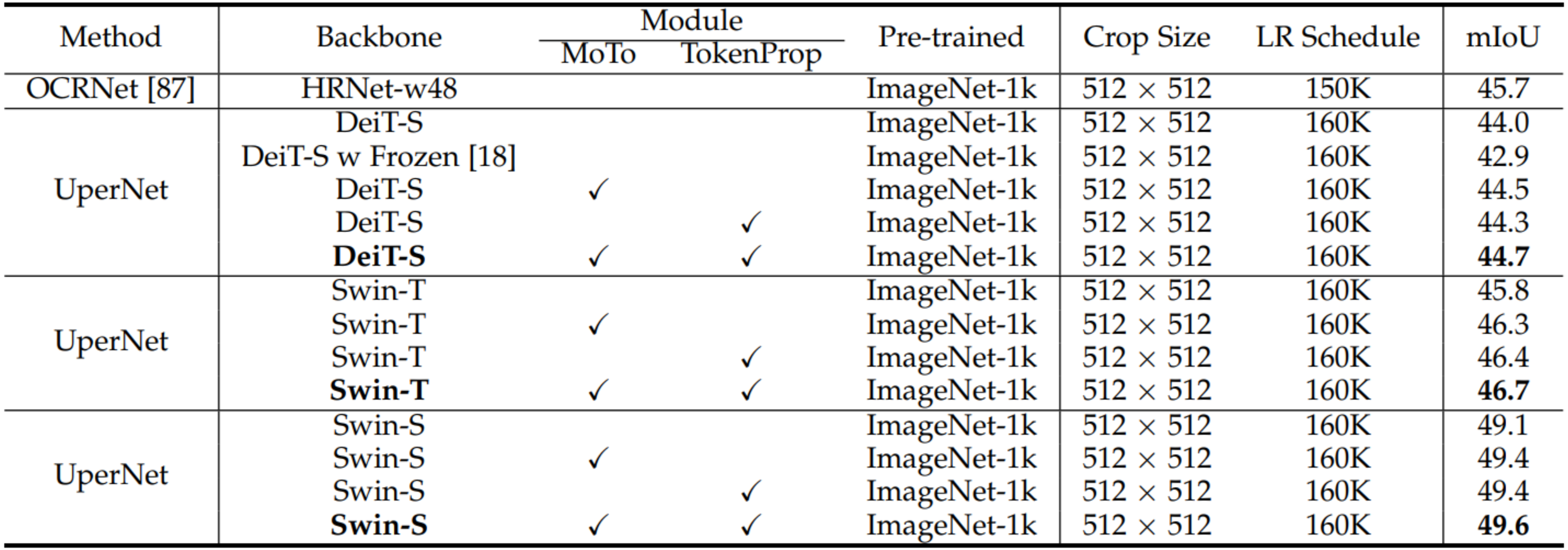
Model Architecture	w MoTO	w TokenProp	Accuracy
DeiT-S	-	-	79.8
Ours	✓	✓	**82.6**
T2T-ViT-14	-	-	81.5
Ours	✓	✓	**82.8**
Swin-T	-	-	81.2
Ours	✓	✓	**82.9**

We follow the original training hyper-parameters in, while only leverage a limited portion of ImageNet-1k training set and 100 epochs for training.

Training Dataset	Percentage (%)	Val Top-1 Accuracy		
		Baseline	w Ours	Δ
ImageNet-1k	50	73.7	**75.1**	+1.4%
	40	71.6	**73.2**	+1.6%
	20	61.2	**63.9**	+2.7%
	10	43.5	**46.5**	+2.9%

4.Experiments - Downstream Tasks

Downstream performance of semantic segmentation on ADE20K dataset.



Method	Backbone	Module		Pre-trained	Crop Size	LR Schedule	mIoU
		MoTo	TokenProp				
OCRNet [87]	HRNet-w48			ImageNet-1k	512 × 512	150K	45.7
UperNet	DeiT-S			ImageNet-1k	512 × 512	160K	44.0
	DeiT-S w Frozen [18]			ImageNet-1k	512 × 512	160K	42.9
	DeiT-S	✓		ImageNet-1k	512 × 512	160K	44.5
	DeiT-S		✓	ImageNet-1k	512 × 512	160K	44.3
	DeiT-S	✓	✓	ImageNet-1k	512 × 512	160K	**44.7**
UperNet	Swin-T			ImageNet-1k	512 × 512	160K	45.8
	Swin-T	✓		ImageNet-1k	512 × 512	160K	46.3
	Swin-T		✓	ImageNet-1k	512 × 512	160K	46.4
	Swin-T	✓	✓	ImageNet-1k	512 × 512	160K	**46.7**
UperNet	Swin-S			ImageNet-1k	512 × 512	160K	49.1
	Swin-S	✓		ImageNet-1k	512 × 512	160K	49.4
	Swin-S		✓	ImageNet-1k	512 × 512	160K	49.4
	Swin-S	✓	✓	ImageNet-1k	512 × 512	160K	**49.6**

4.Experiments - Downstream Tasks

Downstream performance of object detection and instance segmentation on COCO 2017.

Framework	Backbone	Pre-trained	LR Schedule	Box mAP	Mask mAP
RetineNet [88]	PVTv2-b1	ImageNet-1k	1x	41.2	-
	PVTv2-b1 w MoTo	ImageNet-1k	1x	41.8	-
	PVTv2-b1 w TokenProp	ImageNet-1k	1x	41.5	-
	PVTv2-b1 w Both	ImageNet-1k	1x	**42.0**	-
Mask R-CNN [89]	PVTv2-b1	ImageNet-1k	1x	41.8	38.8
	PVTv2-b1 w MoTo	ImageNet-1k	1x	42.4	39.1
	PVTv2-b1 w TokenProp	ImageNet-1k	1x	42.3	39.4
	PVTv2-b1 w Both	ImageNet-1k	1x	**42.9**	**39.4**
Mask R-CNN [89]	Swin-T	ImageNet-1k	1x	43.7	39.8
	Swin-T w MoTo	ImageNet-1k	1x	44.1	40.1
	Swin-T w TokenProp	ImageNet-1k	1x	44.2	40.0
	Swin-T w Both	ImageNet-1k	1x	**44.6**	**40.4**
Cascade Mask R-CNN [90]	Swin-T	ImageNet-1k	1x	48.1	41.7
	Swin-T w MoTo	ImageNet-1k	1x	48.7	42.1
	Swin-T w TokenProp	ImageNet-1k	1x	48.6	41.9
	Swin-T w Both	ImageNet-1k	1x	**49.1**	**42.4**

5. Conclusion and Future Work

[Conclusion]
We demonstrate the **important role tokenizer plays in vision transformers**.

Based on the “trade-off” perspective that formulates prominent structural designs, we propose to incorporate **better normalization and objective for tokenizers.**

Extensive experimental results manifest their **effectiveness across different transformer structures**.

Our findings further indicate that **proper generative supervisory signals help improve** discriminative performance.

[Future Work]
On self-supervised learning with MAE and BEiT validates the potential of reconstruction objectives.

Generalizing such generative signals to different tasks remains an open problem.

Thanks

Any Questions?

You can send mail to
Susang Kim(healess1@gmail.com)