Paper Review

MetaFormer Is Actually What You Need for Vision

Weihao Yu, et al. Sea AI Lab

CVPR 2022(Oral)

Industrial AI Research, POSCO DX

Vision Lab Susang Kim

Contents

1.Introduction - Convolution

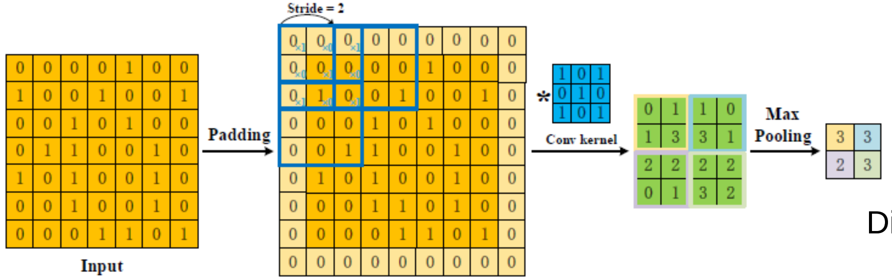
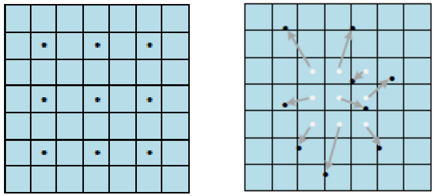


Fig. 1. Procedure of a two-dimensional CNN



Dilated & Deformable convolution kernel.

CNN has been making brilliant achievements, which have become one of the most representative neural networks.

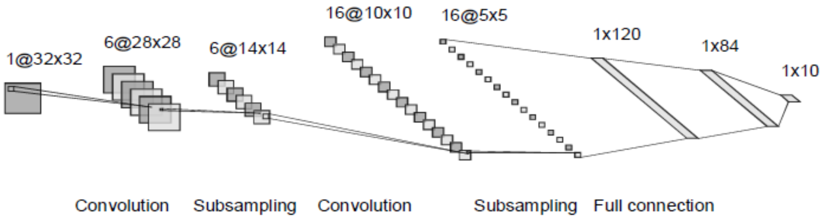
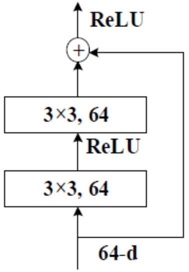
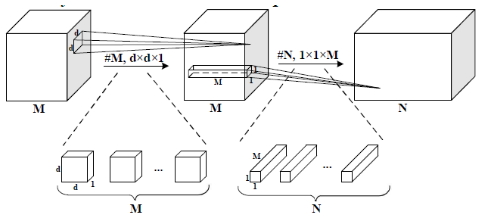
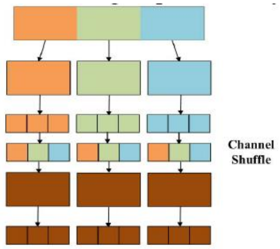


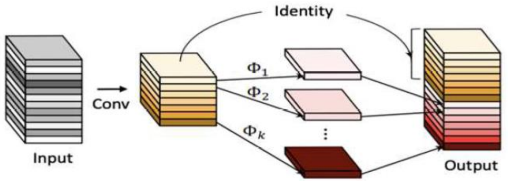
Fig. 5. Architecture of LeNet-5

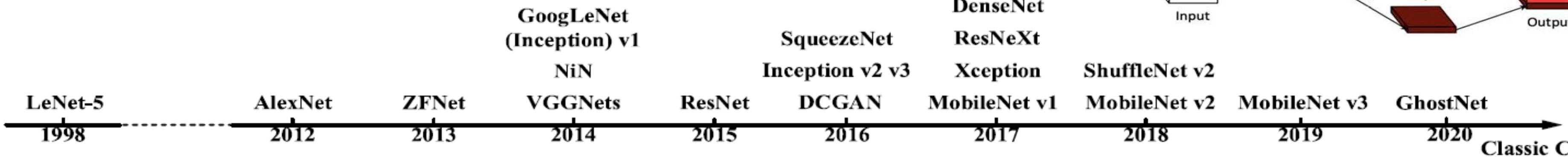






LeCun et al. proposed LeNet-5 in 1998, CNN trained with the backpropagation algorithm.



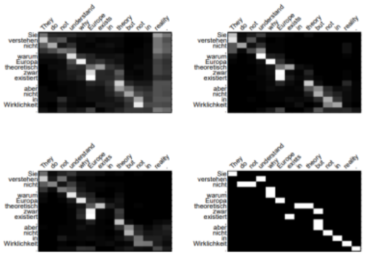


Classic CNN structures

Li, Zewen, et al. "A survey of convolutional neural networks: analysis, applications, and prospects." *IEEE transactions on neural networks* (2021)

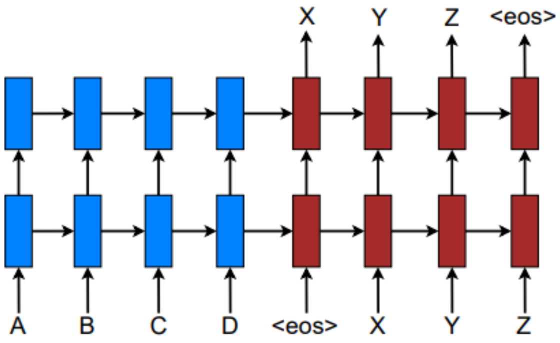
1.Introduction - Attentional Mechanism (Neural machine translation)

Neural machine translation a stacking recurrent architecture for translating a source sequence.



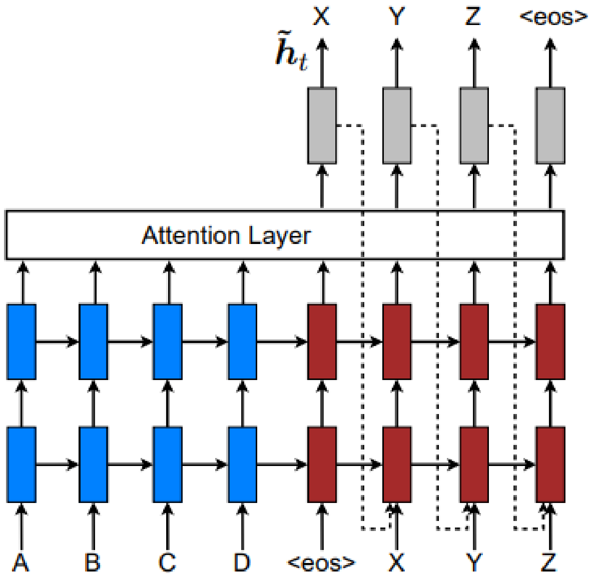
Attention Weights(Hard/Soft)

I like to order a pizza because I’m hungry.



나는 지금 배가 고파서 피자 주문하고 싶다.

I like to order a pizza because I’m hungry.



stacked multiple layers of an RNN(Attentional vectors.

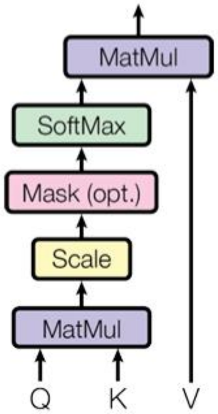
나는 지금 배가 고파서 피자 주문하고 싶다

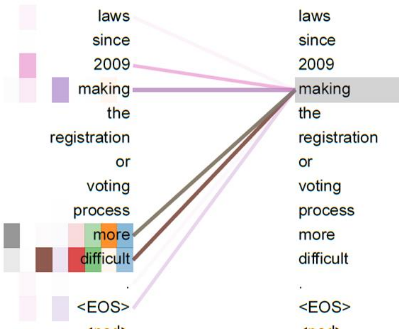
Luong, Thang et al. “Effective Approaches to Attention-based Neural Machine Translation.” EMNLP 2015.

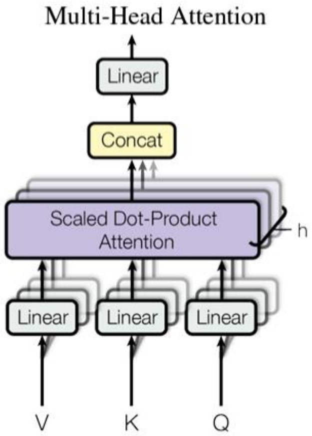
1.Introduction – Self-Attention + Multi-Head Attention

Learning long-range dependencies is a key challenge in many sequence transduction tasks(RNN). As side benefit, self-attention could yield more interpretable models. We inspect attention distributions from our models.

The first sequence transduction model based **entirely on attention**, replacing the recurrent layers most commonly used in encoder-decoder architectures with multi-headed self-attention.







$$\mathrm{Attention}(Q,K,V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

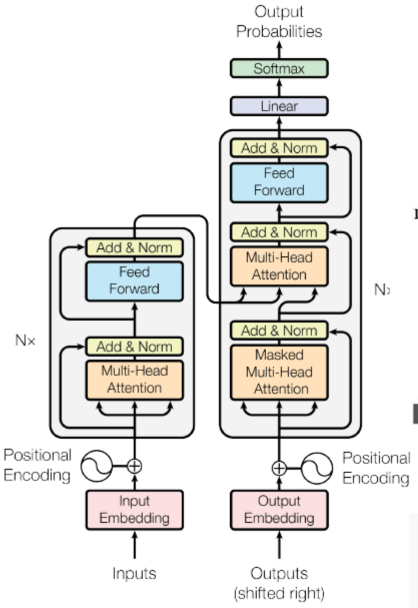
$$\mathrm{MultiHead}(Q,K,V) = \mathrm{Concat}(\mathrm{head_1}, ..., \mathrm{head_h})W^O$$
$$\text{where } \mathrm{head_i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

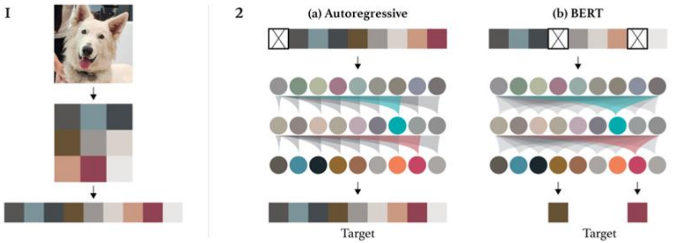
Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

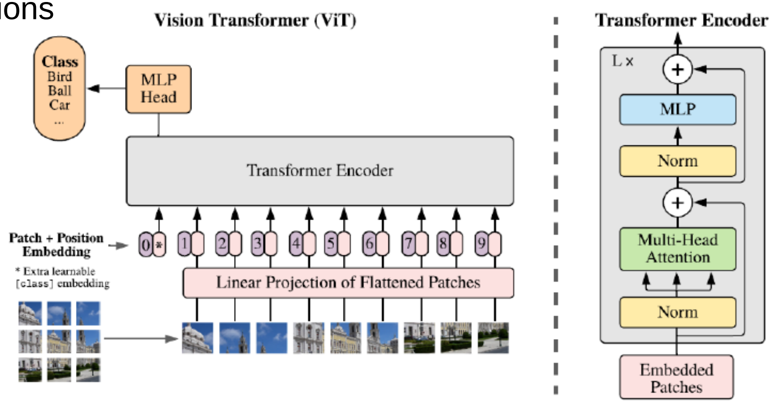
A. Vaswani et al. “Attention is All you Need.”, NeurIPS, 2017.

1.Introduction - Transformer



Both GPT(Generative Pre-trained **Transformer**) and BERT(Bidirectional Encoder Representations **Transformer**) models are based on the Transformer architecture





2017.6 | Transformer
Solely based on attention mechanism, the Transformer is proposed and shows great performance on NLP tasks.

2018.10 | BERT
Pre-training transformer models begin to be dominated in the field of NLP.

2020.5 | GPT-3
A huge transformer with 170B parameters, takes a big step towards general NLP model.

2020.5 | DETR
A simple yet effective framework for high-level vision by viewing object detection as a direct set prediction problem.

2020.7 | iGPT
The transformer model for NLP can also be used for image pre-training.

2020.10 | ViT
Pure transformer architectures work well for visual recognition.

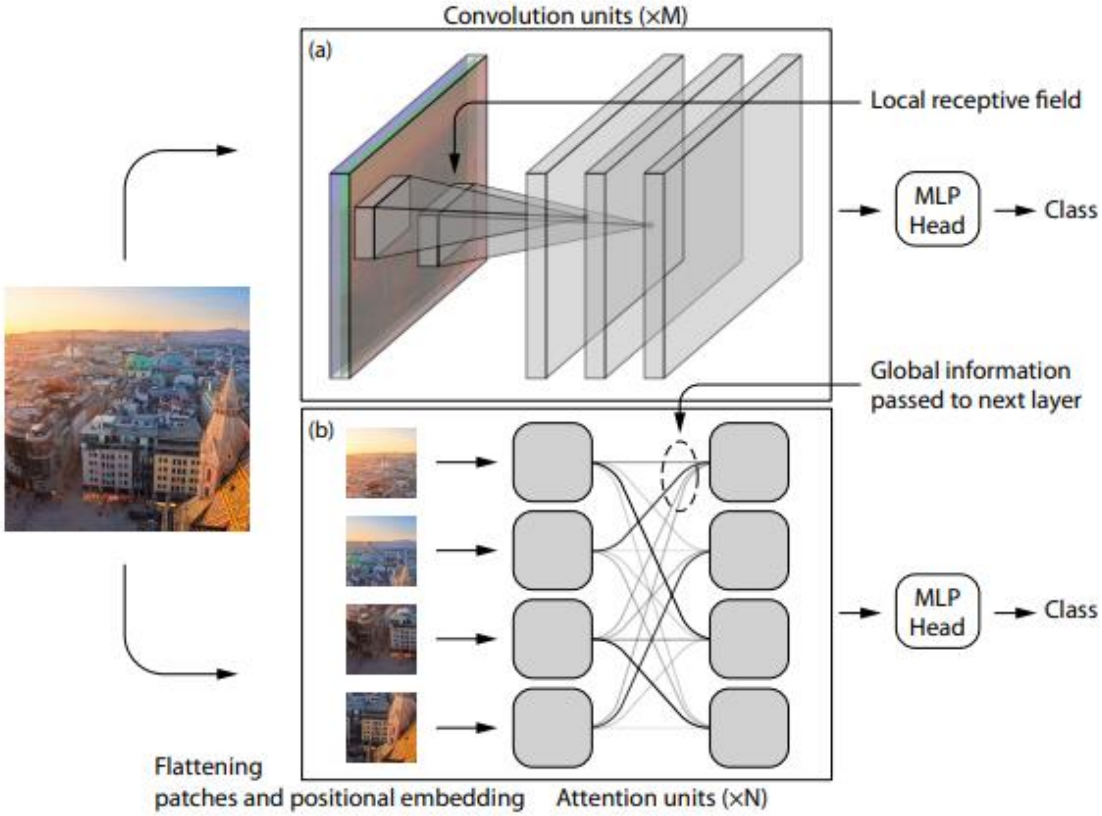
End of 2020 | IPT/SETR/CLIP
Applications of transformer model on low-level vision, segmentation and multimodality tasks, respectively.

2021 | ViT Variants
Variants of ViT models, e.g., DeiT, PVT, TNT, and Swin.

Han, Kai, et al. "A survey on vision transformer." IEEE TPAMI 2022.

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

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

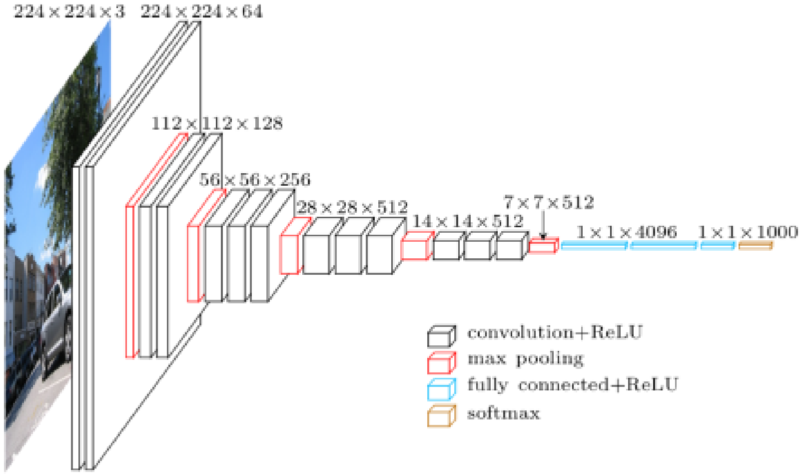


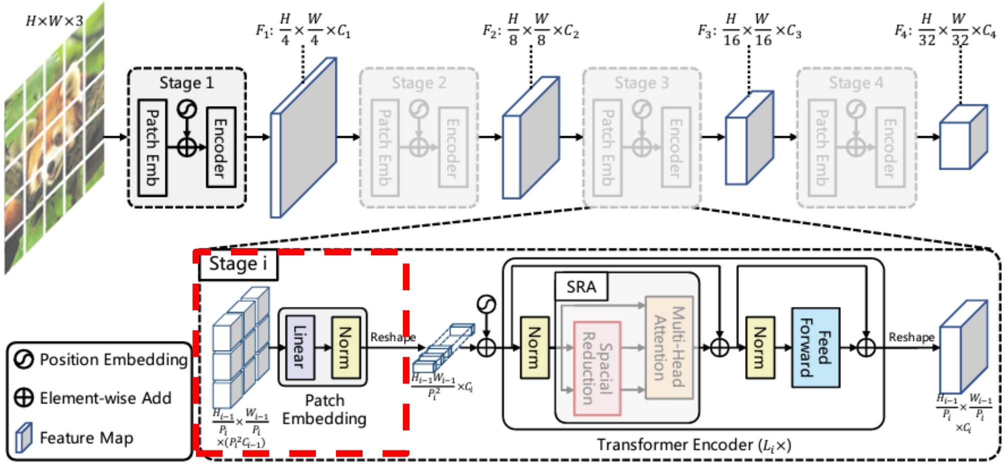
Convolution is efficient in memory and compute.
Local connectivity can lead to loss of global context.
Bad at long sequences(Need to stack many conv layers for outputs to "see" the whole sequence).

Transformers are flexible and attend to information at various distances away from Patch.
Good at long sequences
- output sees "all" inputs.
Dynamic w.r.t input
- output "sees" inputs adaptively.
Very memory-intensive

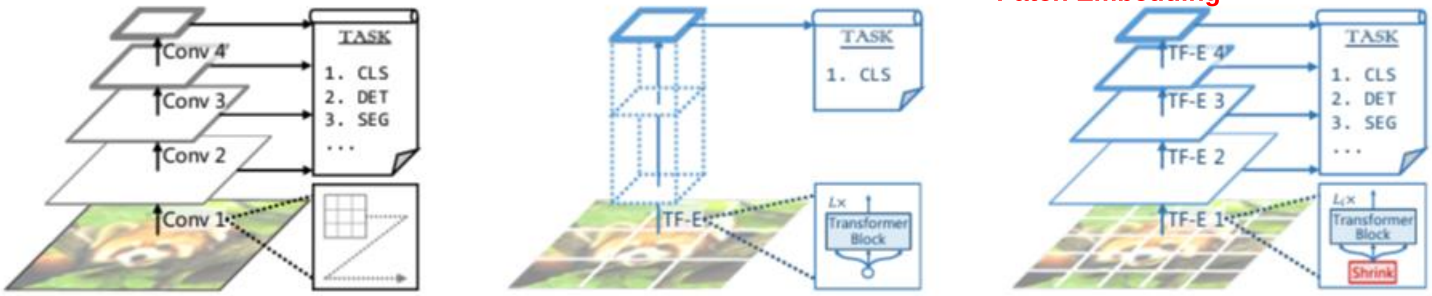
S Tuli et al., "Are Convolutional Neural Networks or Transformers more like human vision?," CogSci, 2021.

1.Introduction - Convolution vs Transformer (Architecture)

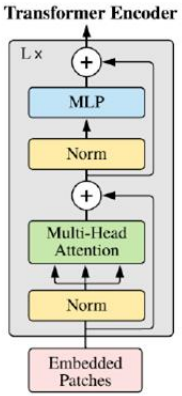




Patch Embedding

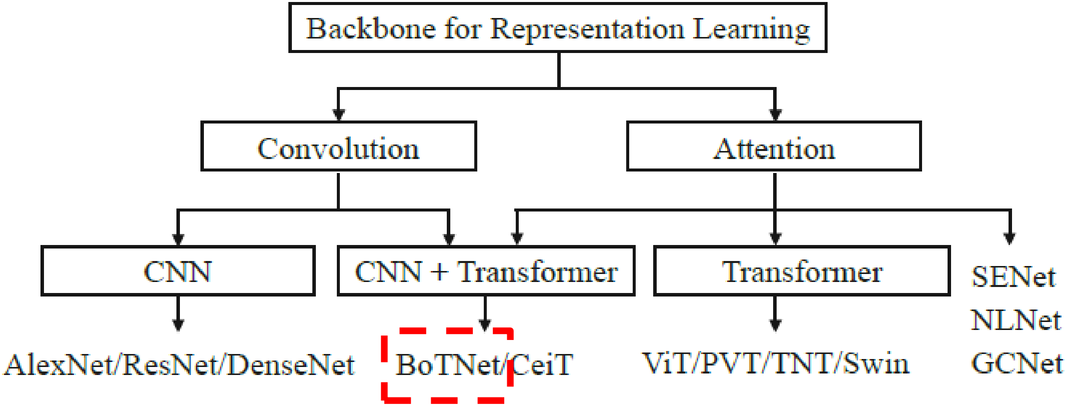


(a) CNNs: VGG [54], ResNet [22], *etc.* (b) Vision Transformer [13] (c) Pyramid Vision Transformer (ours)

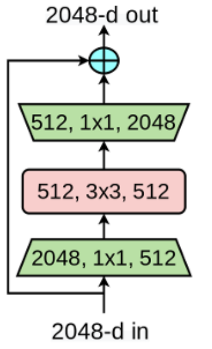


Wang, Wenhai, et al. "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions." ICCV 2021.

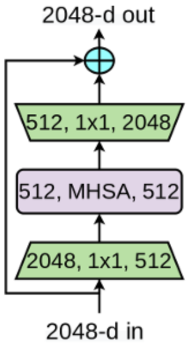
1.Introduction – CNN+Transformer



stage	output	ResNet-50	BoTNet-50
c1	512 × 512	7×7, 64, stride 2	7×7, 64, stride 2
c2	256 × 256	3×3 max pool, stride 2 [1×1, 64 3×3, 64 1×1, 256] ×3	3×3 max pool, stride 2 [1×1, 64 3×3, 64 1×1, 256] ×3
c3	128 × 128	[1×1, 128 3×3, 128 1×1, 512] ×4	[1×1, 128 3×3, 128 1×1, 512] ×4
c4	64 × 64	[1×1, 256 3×3, 256 1×1, 1024] ×6	[1×1, 256 3×3, 256 1×1, 1024] ×6
c5	32 × 32	[1×1, 512 3×3, 512 1×1, 2048] ×3	[1×1, 512 MHSA, 512 1×1, 2048] ×3
# params.		**25.5**×10^6	**20.8**×10^6
M.Adds		**85.4**×10^9	**102.98**×10^9
TPU steptime		**786.5** ms	**1032.66** ms



ResNet Bottleneck



Bottleneck Transformer

By just replacing the spatial convolutions with global self-attention in the final three bottleneck blocks of a ResNet and no other changes.

[Pool, Pool, Pool, Pool] → [Pool, Pool, Pool, Attention]	14.0	1.9	78.3
[Pool, Pool, Pool, Pool] → [Pool, Pool, Attention, Attention]	16.5	2.5	81.0

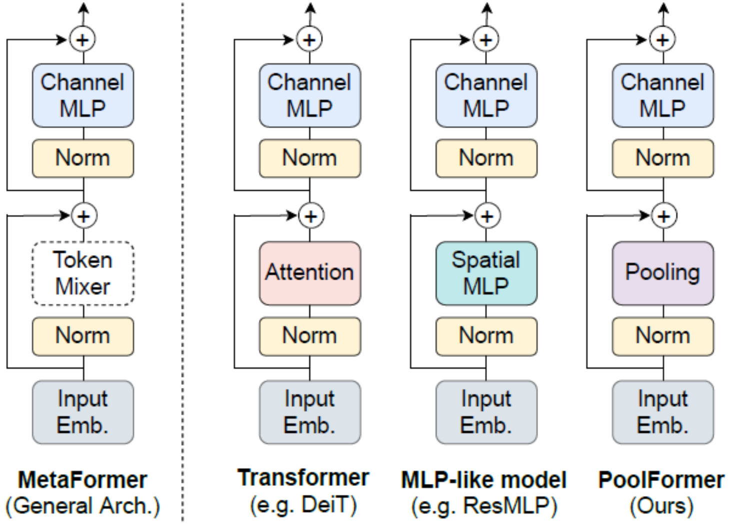
Srinivas, Aravind, et al. "Bottleneck transformers for visual recognition." *CVPR 2021.*

1.Introduction – PoolFormer, CVPR 2022 (Oral)

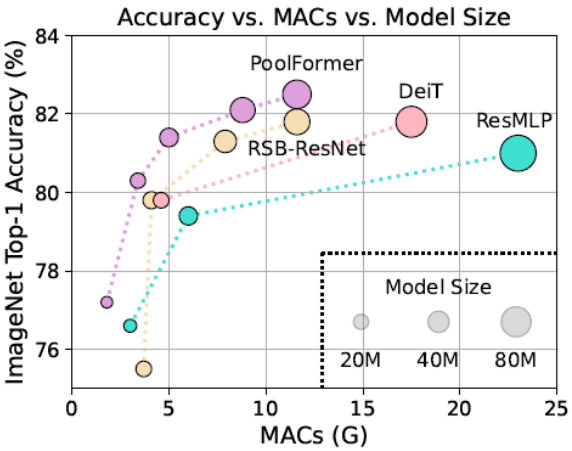
MetaFormer Is Actually What You Need for Vision

Weihao Yu[1,2*] Mi Luo[1] Pan Zhou[1] Chenyang Si[1] Yichen Zhou[1,2]
Xinchao Wang[2] Jiashi Feng[1] Shuicheng Yan[1]
[1]Sea AI Lab [2]National University of Singapore
weihaoyu6@gmail.com {luomi,zhoupan,sicy,zhouyc,fengjs,yansc}@sea.com xinchao@nus.edu.sg

Code: https://github.com/sail-sg/poolformer



(a)



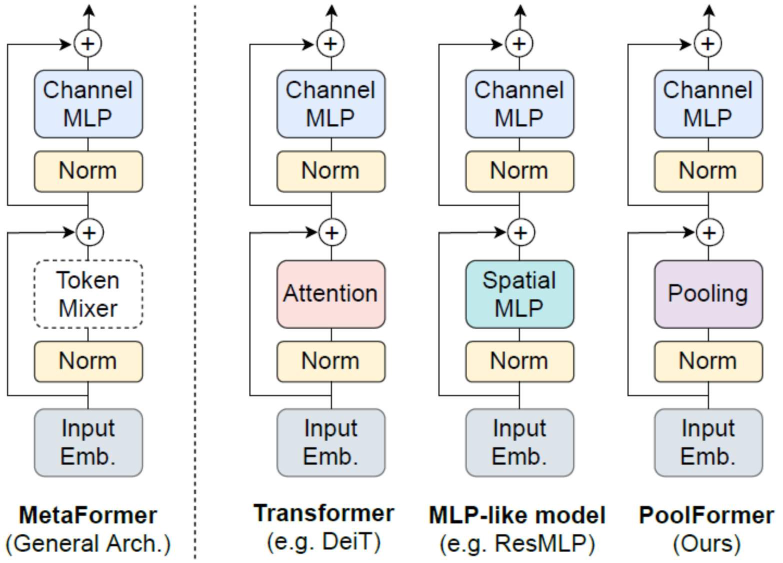
(b)

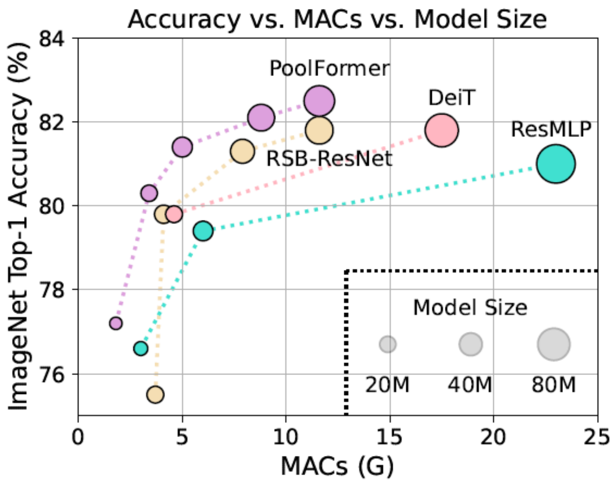
Srinivas, Aravind, et al. "Bottleneck transformers for visual recognition." *CVPR 2021.*

2.Motivation - Token Mixer is all you need?

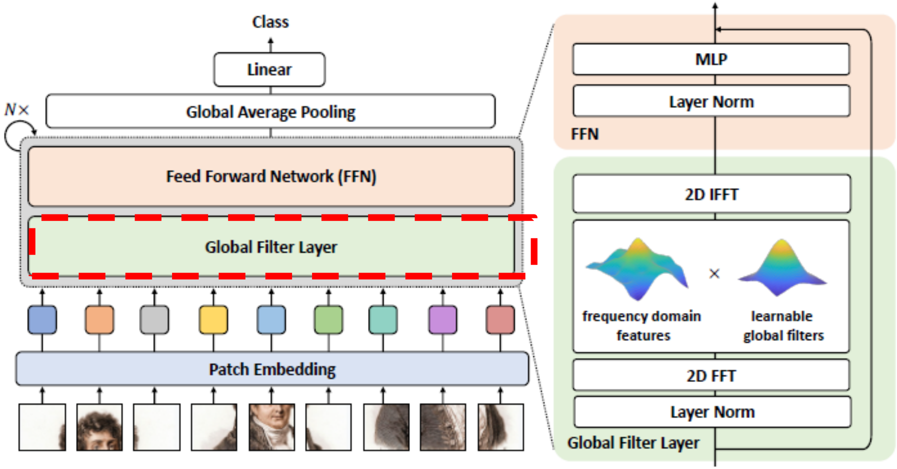
What is the success of Transformers? Our answer is the general architecture MetaFormer.

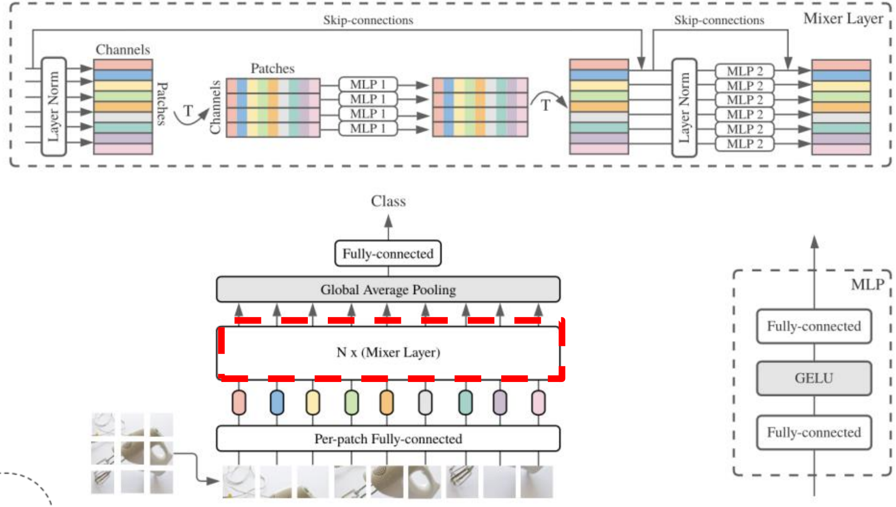
Transformers have shown great potential in computer vision tasks. **A common belief is their attention-based token mixer module** contributes most to their competence.
Based on this observation, we hypothesize that **the general architecture of the Transformers, instead of the specific token mixer module,** is more essential to the model's performance.
"MetaFormer", a general architecture abstracted from Transformers without specifying the token mixer.

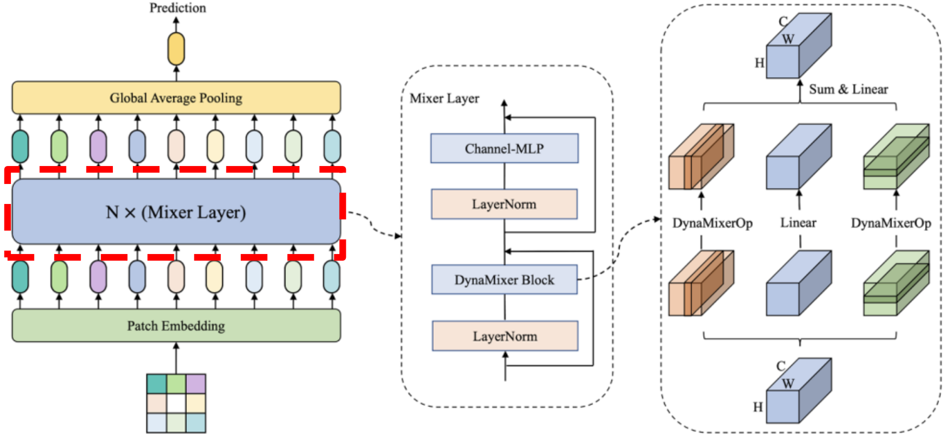


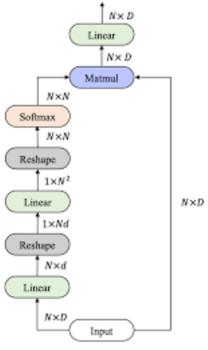


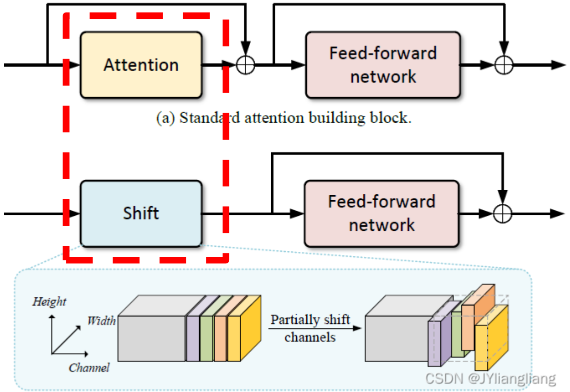
2.Motivation -Token Mixer is all you need?





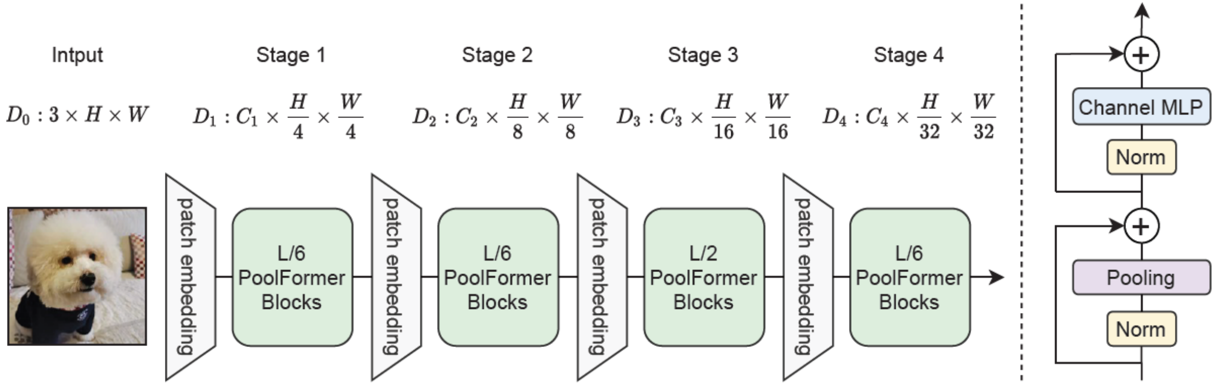






GFNet(NeurIPS 2021), MLP Mixer(NeurIPS 2021), ShiftViT(AAAI2022), DynaMixer(ICML 2022)

3.Method - PoolFormer



(a) Overal framework with L PoolFormer blocks

(b) PoolFormer block

$$Z = \sigma(\mathrm{Norm}(Y)W_1)W_2 + Y,$$

$$Y = \mathrm{TokenMixer}(\mathrm{Norm}(X)) + X,$$

$$X = \mathrm{InputEmb}(I),$$

Algorithm 1 Pooling for PoolFormer, PyTorch-like Code

```
import torch.nn as nn

class Pooling(nn.Module):
   def __init__(self, pool_size=3):
      super().__init__()
      self.pool = nn.AvgPool2d(
         pool_size, stride=1,
         padding=pool_size//2,
         count_include_pad=False,
      )
   def forward(self, x):
      """
      [B, C, H, W] = x.shape
      Subtraction of the input itself is added
      since the block already has a
      residual connection.
      """
      return self.pool(x) - x
```

$$T'_{:,i,j} = \frac{1}{K \times K} \sum_{p,q=1}^{K} T_{:,i+p-\frac{K+1}{2},i+q-\frac{K+1}{2}} - T_{:,i,j},$$

K is the pooling size $T \in \mathbb{R}^{C \times H \times W}$

We exploit an embarrassingly simple non-parametric operator, pooling, to conduct extremely basic token mixing.

3.Method - Configurations of different PoolFormer models.

Stage	#Tokens	Layer Specification		PoolFormer				
				S12	S24	S36	M36	M48
1	$\frac{H}{4} \times \frac{W}{4}$	Patch Embedding	Patch Size	7×7, stride 4				
			Embed. Dim.	64			96	
		PoolFormer Block	Pooling Size	3×3, stride 1				
			MLP Ratio	4				
			# Block	2	4	6	6	8
2	$\frac{H}{8} \times \frac{W}{8}$	Patch Embedding	Patch Size	3×3, stride 2				
			Embed. Dim.	128			192	
		PoolFormer Block	Pooling Size	3×3, stride 1				
			MLP Ratio	4				
			# Block	2	4	6	6	8
3	$\frac{H}{16} \times \frac{W}{16}$	Patch Embedding	Patch Size	3×3, stride 2				
			Embed. Dim.	320			384	
		PoolFormer Block	Pooling Size	3×3, stride 1				
			MLP Ratio	4				
			# Block	6	12	18	18	24
4	$\frac{H}{32} \times \frac{W}{32}$	Patch Embedding	Patch Size	3×3, stride 2				
			Embed. Dim.	512			768	
		PoolFormer Block	Pooling Size	3×3, stride 1				
			MLP Ratio	4				
			# Block	2	4	6	6	8
Parameters (M)				11.9	21.4	30.8	56.1	73.4
MACs (G)				1.8	3.4	5.0	8.8	11.6

There are two groups of embedding size
1) small-sized models with embedding dimensions of 64, 128, 320, and 512 responding to the four stages
2) medium-sized models with embedding dimensions 96, 192, 384, and 768.

4.Experiments – ImageNet-1K

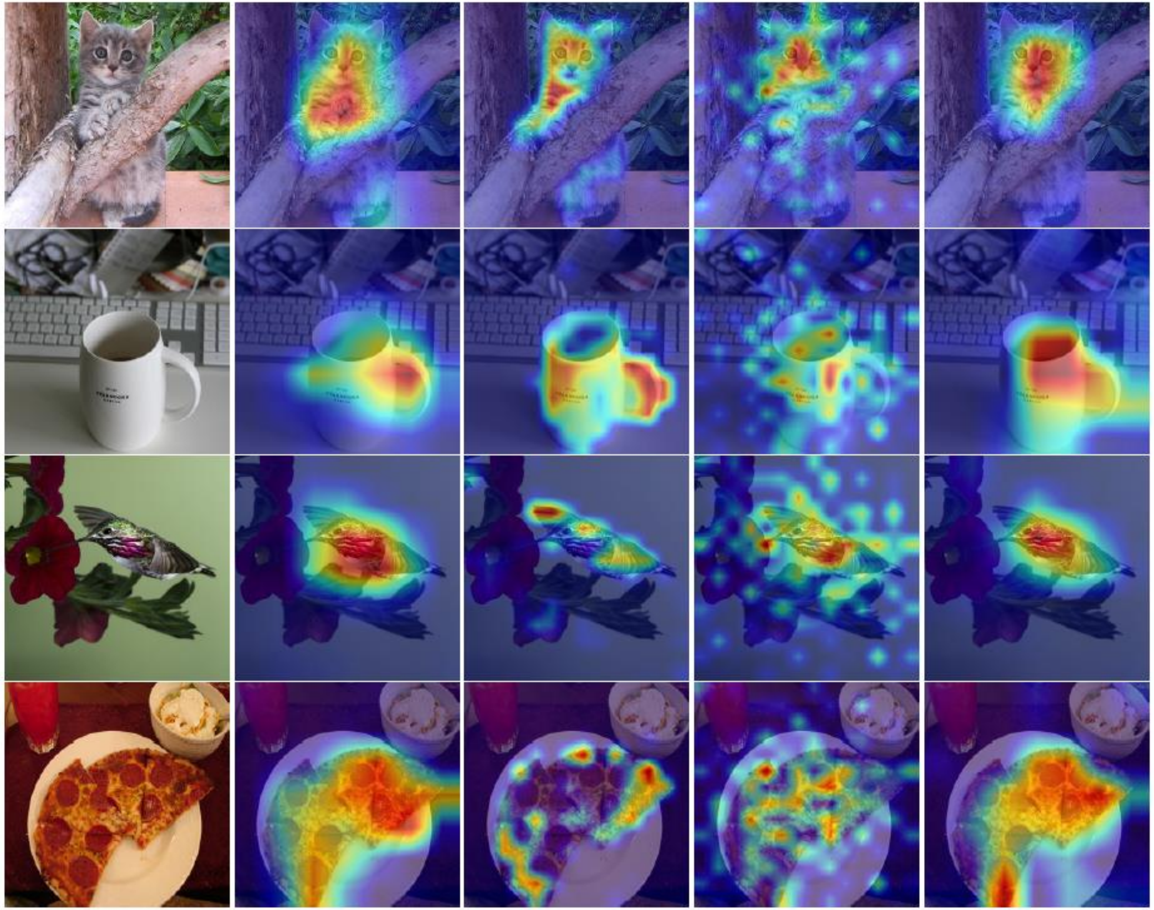
General Arch.	Token Mixer	Outcome Model	Image Size	Params (M)	MACs (G)	Top-1 (%)
Convolutional Neural Netowrks	—	RSB-ResNet-18 [24, 59]	224	12	1.8	70.6
		RSB-ResNet-34 [24, 59]	224	22	3.7	75.5
		RSB-ResNet-50 [24, 59]	224	26	4.1	79.8
		RSB-ResNet-101 [24, 59]	224	45	7.9	81.3
		RSB-ResNet-152 [24, 59]	224	60	11.6	81.8
MetaFormer	Attention	ViT-B/16* [17]	224	86	17.6	79.7
		ViT-L/16* [17]	224	307	63.6	76.1
		DeiT-S [53]	224	22	4.6	79.8
		DeiT-B [53]	224	86	17.5	81.8
		PVT-Tiny [57]	224	13	1.9	75.1
		PVT-Small [57]	224	25	3.8	79.8
		PVT-Medium [57]	224	44	6.7	81.2
		PVT-Large [57]	224	61	9.8	81.7
	Spatial MLP	MLP-Mixer-B/16 [51]	224	59	12.7	76.4
		ResMLP-S12 [52]	224	15	3.0	76.6
		ResMLP-S24 [52]	224	30	6.0	79.4
		ResMLP-B24 [52]	224	116	23.0	81.0
		Swin-Mixer-T/D24 [36]	256	20	4.0	79.4
		Swin-Mixer-T/D6 [36]	256	23	4.0	79.7
		Swin-Mixer-B/D24 [36]	224	61	10.4	81.3
		gMLP-S [35]	224	20	4.5	79.6
		gMLP-B [35]	224	73	15.8	81.6
	Pooling	PoolFormer-S12	224	12	1.8	77.2
		PoolFormer-S24	224	21	3.4	80.3
		PoolFormer-S36	224	31	5.0	81.4
		PoolFormer-M36	224	56	8.8	82.1
		PoolFormer-M48	224	73	11.6	82.5

All these models are only trained on the ImageNet-1K training set and the accuracy on the validation set is reported.

Surprisingly, despite **the simple pooling token mixer**, PoolFormers can still achieve highly competitive performance compared with CNNs and other MetaFormer.

PoolFormers can still achieve highly competitive performance compared with CNNs and other MetaFormer like models.,

4.Experiments – Grad-CAM

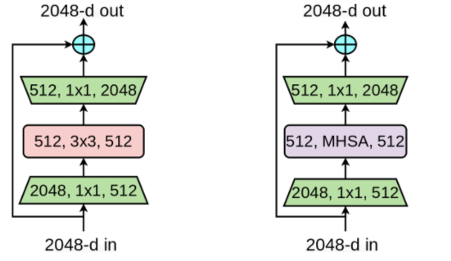


Input RSB-ResNet-50 [59] DeiT-small [53] ResMLP-S24 [52] PoolFormer-S24

Grad-CAM activation maps of the models trained on ImageNet-1K. The visualized images are from validation set.

4.Experiments – Ablation Study

Ablation	Variant	Params (M)	MACs (G)	Top-1 (%)
Baseline	None (PoolFormer-S12)	11.9	1.8	77.2
Token mixers	Pooling → Identity mapping	11.9	1.8	74.3
	Pooling → Global random matrix* (extra 21M frozen parameters)	11.9	3.3	75.8
	Pooling → Depthwise Convolution [9, 38]	11.9	1.8	78.1
	Pooling size 3 → 5	11.9	1.8	77.2
	Pooling size 3 → 7	11.9	1.8	77.1
	Pooling size 3 → 9	11.9	1.8	76.8
Normalization	Modified Layer Normalization† → Layer Normalization [1]	11.9	1.8	76.5
	Modified Layer Normalization† → Batch Normalization [28]	11.9	1.8	76.4
	Modified Layer Normalization† → None	11.9	1.8	46.1
Activation	GELU [25] → ReLU [41]	11.9	1.8	76.4
	GELU → SiLU [18]	11.9	1.8	77.2
Other components	Residual connection [25] → None	11.9	1.8	0.1
	Channel MLP → None	2.5	0.2	5.7
Hybrid Stages	[Pool, Pool, Pool, Pool] → [Pool, Pool, Pool, Attention]	14.0	1.9	78.3
	[Pool, Pool, Pool, Pool] → [Pool, Pool, Attention, Attention]	16.5	2.5	81.0
	[Pool, Pool, Pool, Pool] → [Pool, Pool, Pool, SpatialFC]	11.9	1.8	77.5
	[Pool, Pool, Pool, Pool] → [Pool, Pool, SpatialFC, SpatialFC]	12.2	1.9	77.9



ResNet Bottleneck
Bottleneck Transformer

# Epochs	300 (default)	400	500	1000	1500	2000	2500	3000
PoolFormer-S12	77.2	77.5	77.9	78.4	78.6	78.8	78.8	78.8

Table 7. **Performance of PoolFormer trained for different numbers of epochs.**

We observe that PoolFormer obtains saturated after around 2000 epochs with a top-1 accuracy improvement of 1.8%.

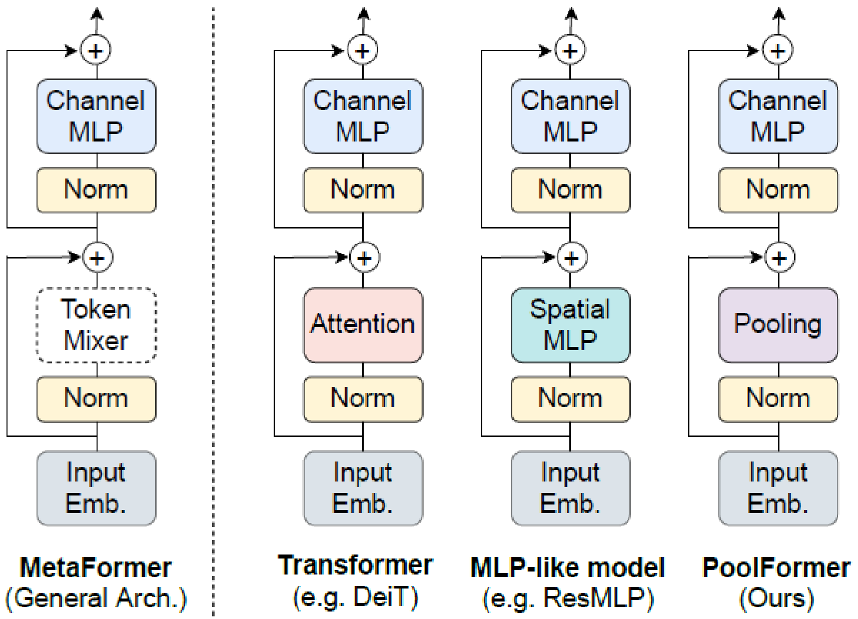
5.Conclusion and future work

We deliberately specify token mixer as extremely simple pooling for MetaFormer.
It is found that the derived PoolFormer model can achieve competitive performance on different vision tasks, which well supports that **"MetaFormer is actually what you need for vision"**.

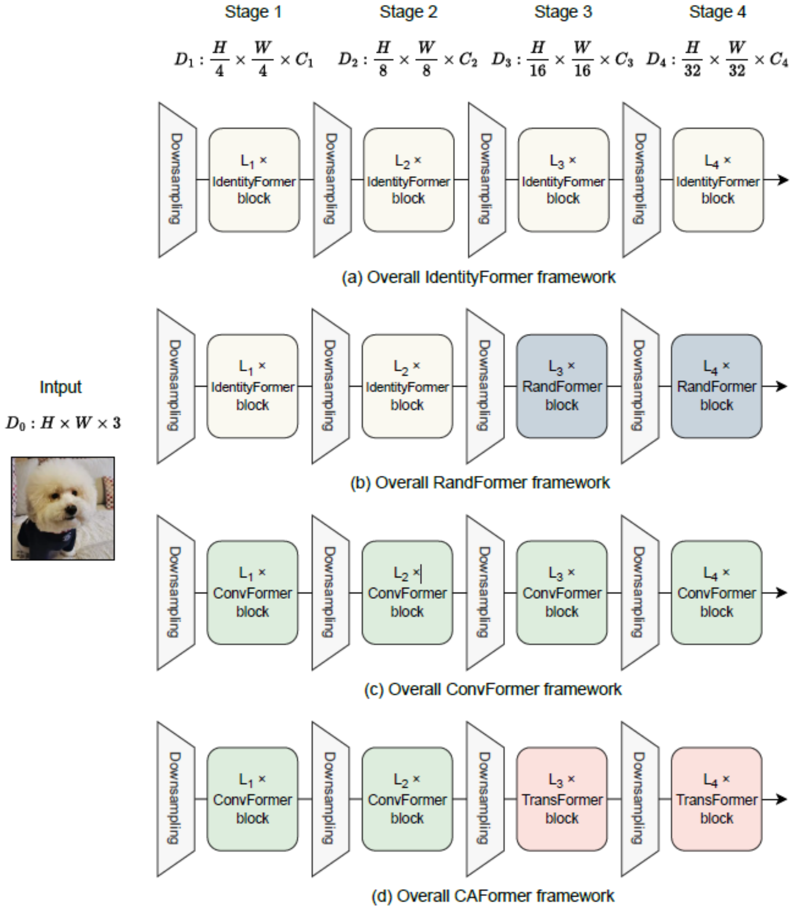
We will further evaluate PoolFormer under more different learning settings, such as **self-supervised learning and transfer learning.** Moreover, it is interesting to see whether PoolFormer still works on **NLP tasks**

We hope that this work can inspire more future research devoted **to improving the fundamental architecture** MetaFormer instead of paying too much attention to the token mixer modules.

The PoolFormer can readily serve as a good starting baseline for future **MetaFormer architecture design**



5.MetaFormer Baselines for Vision (Arxiv – 2022.12.22)



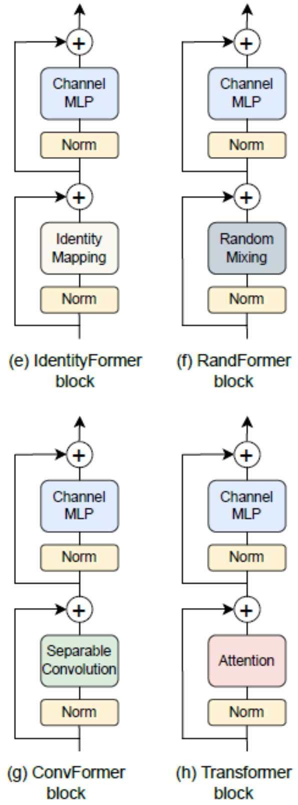
(a) Overall IdentityFormer framework

(b) Overall RandFormer framework

(c) Overall ConvFormer framework

(d) Overall CAFormer framework

Overall frameworks of IdentityFormer, RandFormer, ConvFormer and CAFormer.



(e) IdentityFormer block

(f) RandFormer block

(g) ConvFormer block

(h) Transformer block

Variant	Top-1 (%) ConvFormer-S18	Top-1 (%) CAFormer-S18
Baseline	83.0	83.6
StarReLU → ReLU [49]	82.1 (-0.9)	82.9 (-0.7)
StarReLU → Squared ReLU [63]	82.6 (-0.4)	83.4 (-0.2)
StarReLU → GELU [25]	82.7 (-0.3)	83.4 (-0.2)

$$\mathrm{StarReLU}(x) = s \cdot (\mathrm{ReLU}(x))^2 + b,$$

$$\mathrm{StarReLU}(x) = \frac{(\mathrm{ReLU}(x))^2 - \mathrm{E}\left((\mathrm{ReLU}(x))^2\right)}{\sqrt{\mathrm{Var}\left((\mathrm{ReLU}(x))^2\right)}} = \frac{(\mathrm{ReLU}(x))^2 - 0.5}{\sqrt{1.25}}$$
$$\approx 0.8944 \cdot (\mathrm{ReLU}(x))^2 - 0.4472.$$

$$X = \mathrm{InputEmbedding}(I).$$

$$X' = X + \mathrm{TokenMixer}\left(\mathrm{Norm}_1(X)\right),$$
$$X'' = X' + \sigma\left(\mathrm{Norm}_2(X')W_1\right)W_2,$$

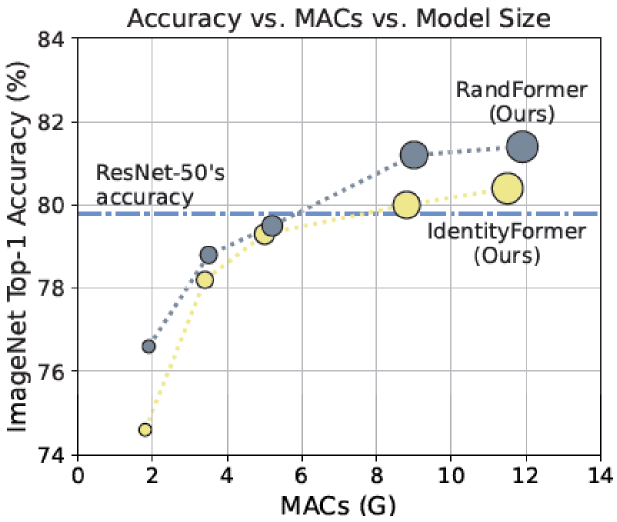
$$\mathrm{IdentityMapping}(X) = X.$$

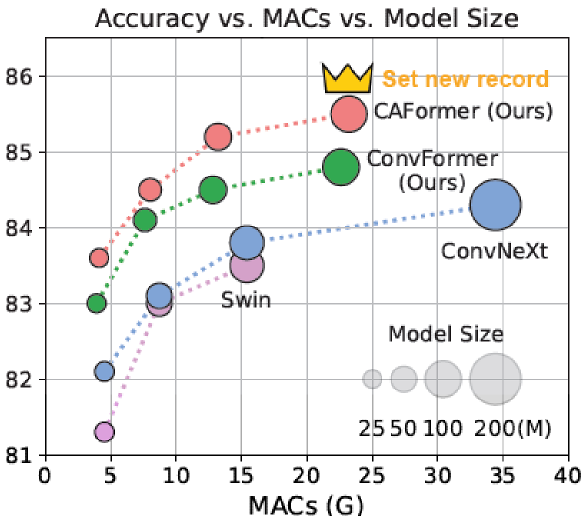
$$\mathrm{RandomMixing}(X) = XW_R,$$

$$\mathrm{Convolutions(X)} = \mathrm{Conv}_{\mathrm{pw2}}(\mathrm{Conv}_{\mathrm{dw}}(\sigma(\mathrm{Conv}_{\mathrm{pw1}}(X)))),$$

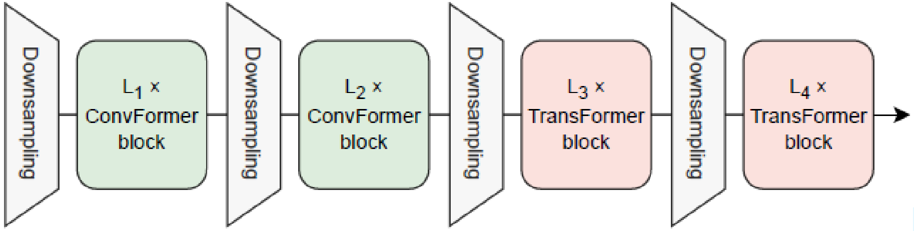
Yu, Weihao, et al. "Metaformer baselines for vision." arXiv 2022.

5.MetaFormer Baselines for Vision (Arxiv – 2022.12.22)





CAFormer sets new record on ImageNet-1K. By simply applying depthwise separable convolutions as token mixer in the bottom stages and vanilla self-attention in the top stages, the resulting model CAFormer sets a new record on ImageNet-1K

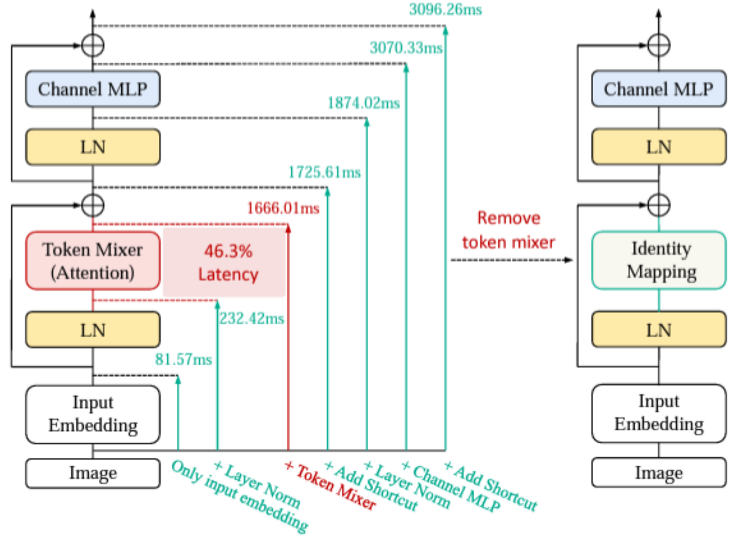


(d) Overall CAFormer framework

Swin-B [45]	✓	Attn	88	15.4	83.5
CSwin-B [16]	✓	Attn	78	15.0	84.2
MViTv2-B [40]	✓	Attn	52	10.2	84.4
CoAtNet-2 [12]	✓	Conv + Attn	75	15.7	84.1
MaxViT-S [71]	✓	Conv + Attn	69	11.7	84.5
iFormer-L [62]	✓	Conv + Attn	87	14.0	84.8
CAFormer-M36	✓	Conv + Attn	56	13.2	**85.2**

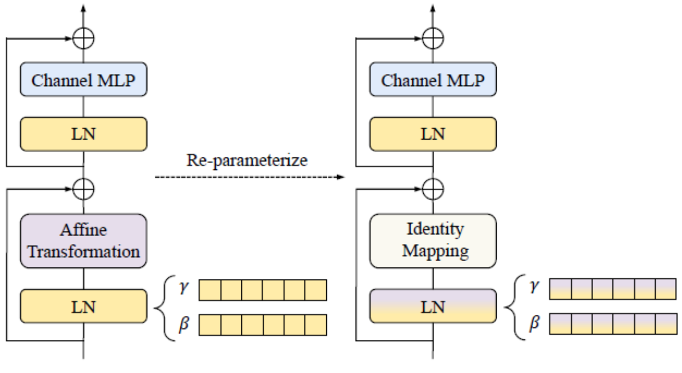
Yu, Weihao, et al. "Metaformer baselines for vision." arXiv 2022.

5.RIFormer - Removing Token Mixer (CVPR 2023)



(a) Latency analysis of ViT-B (b) Remove token mixer with heavy latency

RepIdentityFormer base on the re-parameterizing idea, to study the token mixer free model architecture.



(a) RepIdentityFormer Training (b) RepIdentityFormer Inference

Figure 2. Structural re-parameterization of a RIFormer block.

Motivated by their considerable latency cost. We observe that appropriate optimization strategy can effectively help a **token mixer-free model** learn useful knowledge from another model.

● PoolFormer-S12 [52]	224	12	1.8	4160.18	77.2
● PoolFormer-S24 [52]	224	21	3.4	2140.20	80.3
● PoolFormer-S36 [52]	224	31	5.0	1440.37	81.4
● PoolFormer-M36 [52]	224	56	8.8	1009.45	82.1
● PoolFormer-M48 [52]	224	73	11.6	761.93	82.5
★ RIFormer-S12$^\diamond$	224	12	1.8	4899.80 (↑ 17.8%)	76.9
★ RIFormer-S24$^\diamond$	224	21	3.4	2530.48 (↑ 18.2%)	80.3
★ RIFormer-S36$^\diamond$	224	31	5.0	1699.94 (↑ 18.0%)	81.3
★ RIFormer-M36$^\diamond$	224	56	8.8	1185.33 (↑ 17.4%)	82.6
★ RIFormer-M48$^\diamond$	224	73	11.6	897.05 (↑ 17.7%)	82.8

Wang, Jiahao, et al. "RIFormer: Keep Your Vision Backbone Effective While Removing Token Mixer.” CVPR 2023.

Thanks

Any Questions?

You can send mail to

Susang Kim(healess1@gmail.com)