Conformer: Local Features Coupling Global Representations for Visual Recognition

Zhiliang Peng[1] Wei Huang[1] Shanzhi Gu[3] Lingxi Xie[2] Yaowei Wang[3]
Jianbin Jiao[1] Qixiang Ye[1,3]
[1]University of Chinese Academy of Sciences, Beijing, China [2]Huawei Inc.
[3]Peng Cheng Laboratory, Shenzhen, China
{pengzhiliang19, huangwei19}@mails.ucas.ac.cn {gushzh, wangyw}@pcl.ac.cn
198808xc@gmail.com {jiaojb, qxye}@ucas.ac.cn

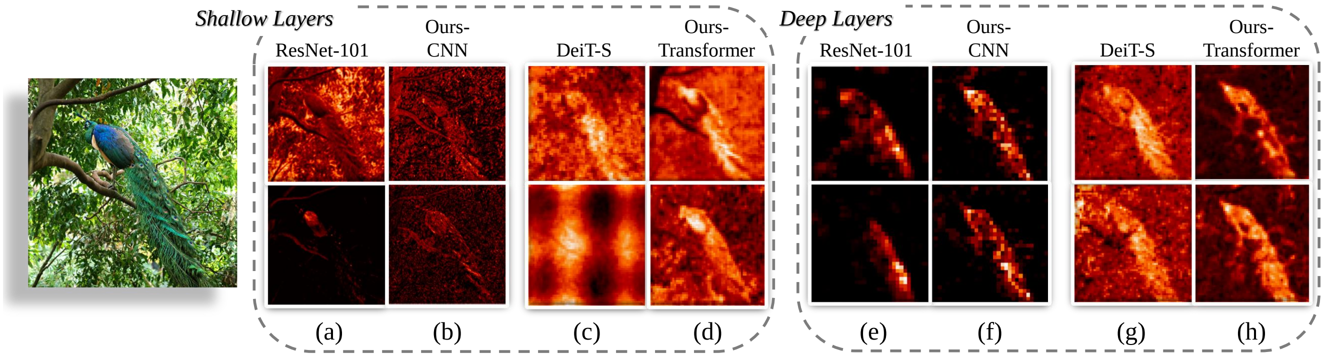


Figure 1: Comparison of feature maps of CNN (ResNet-101) [18], Visual Transformer (DeiT-S) [41], and the proposed Conformer. The patch embeddings in transformer are reshaped to feature maps for visualization. While CNN activates discriminative local regions (*e.g.*, the peacock's head in (a) and tail in (e)), the CNN branch of Conformer takes advantage of global cues from the visual transformer and thereby activates complete object (*e.g.*, full extent of the peacock in (b) and (f)). Compared with CNN, local feature details of the visual transformer are deteriorated (*e.g.*, (c) and (g)). In contrast, the transformer branch of Conformer retains the local feature details from CNN while depressing the background (*e.g.*, the peacock contours in (d) and (h) are more complete than those in (c) and (g)). (Best viewed in color)

Abstract

Within Convolutional Neural Network (CNN), the convolution operations are good at extracting local features but experience difficulty to capture global representations. Within visual transformer, the cascaded self-attention modules can capture long-distance feature dependencies but unfortunately deteriorate local feature details. In this paper, we propose a hybrid network structure, termed Conformer, to take advantage of convolutional operations and self-attention mechanisms for enhanced representation learning. Conformer roots in the Feature Coupling Unit (FCU), which fuses local features and global representations under different resolutions in an interactive fashion. Conformer adopts a concurrent structure so that local features and global representations are retained to the maximum extent. Experiments show that Conformer, under the comparable parameter complexity, outperforms the visual transformer (DeiT-B) by 2.3% on ImageNet. On MSCOCO, it outperforms ResNet-101 by 3.7% and 3.6% mAPs for object detection and instance segmentation, respectively, demonstrating the great potential to be a general backbone network. Code is available at github.com/pengzhiliang/Conformer.

1. Introduction

Convolutional neural networks (CNNs) [28, 36, 39, 18, 47, 21] have significantly advanced computer vision tasks such as image classification, object detection, and instance segmentation. This largely attributes to the convolution operation, which collects local features in a hierarchical fashion as powerful image representations. Despite of the advantage upon local feature extraction, CNNs experience difficulty to capture global representations, *e.g.*, long-distance relationships among visual elements, which are often critical for high-level computer visual tasks. An intuitive solution is enlarging the receptive field, which however could require more intensive yet damaging pooling operations.

Recently, the transformer architecture [42] has been introduced to visual tasks [16, 46, 41, 50, 8, 9, 3, 53, 27]. The ViT method [16] constructs a sequence of tokens by splitting each image to patches with positional embeddings and applies cascaded transformer blocks to extract parameterized vectors as visual representations. Thanks to the self-attention mechanism and Multilayer Perceptron (MLP) structure, the visual transformer reflects complex spatial transforms and long-distance feature dependencies, which constitute global representations. Unfortunately, visual transformers are observed ignoring local feature details which decreases the discriminability between background and foreground, Figs. 1(c) and (g). Improved visual transformers [16, 50] have proposed a tokenization module or leveraged CNN feature maps as input tokens to capture feature neighboring information. Nevertheless, the problem about how to precisely embed local features and global representations to each other remains.

In this paper, we propose a dual network structure, termed Conformer, with the aim to couple CNN-based local features with transformer-based global representations for enhanced representation learning. Conformer consists of a CNN branch and a transformer branch which respectively follow the design of ResNet [18] and ViT [16]. The two branches form a comprehensive combination of local convolution blocks, self-attention modules, and MLP units. During training, the cross entropy losses are used to supervise both the CNN and transformer branches to couple CNN-style and transformer-style features.

Considering the feature misalignment between CNN and transformer features, the Feature Coupling Unit (FCU) is designed as the bridge. On the one hand, to fuse the two-style features, FCU leverages 1×1 convolution to align the channel dimensions, down/up sampling strategies to align feature resolutions, LayerNorm [2] and BatchNorm [24] to align feature values. On the other hand, since CNN and transformer branches tend to capture features of different levels (*e.g.*, local vs. global), FCU is inserted into every block to consecutively eliminate the semantic divergence between them, in an interactive fashion. Such a fusion procedure can greatly enhance the global perception capability of local features and the local details of global representations.

The ability of Conformer in coupling local features and global representations is demonstrated in Fig. 1. While conventional CNNs (*e.g.*, ResNet-101) tend to retain discriminative local regions (*e.g.*, the peacock's head or tail), the CNN branch of Conformer can activate the full object extent, Figs. 1(b) and (f). When solely using the visual transformers, for the weak local features (*e.g.*, blurred object boundaries), it is difficult to distinguish the object from the background, Figs. 1(c) and (g). The coupling of local features and global representations significantly enhances the discriminability of transformer-based features, Figs. 1(d) and (h).

The contributions of this paper include:

- We propose a dual network structure, termed Conformer, which retains local features and global representations to the maximum extent.
- We propose the Feature Coupling Unit (FCU), to fuse convolutional local features with transformer-based global representations in an interactive fashion.
- Under comparable parameter complexity, Conformer outperforms CNNs and visual transformers by significant margins. Conformer inherits the structure and generalization advantages of both CNNs and visual transformers, demonstrating the great potential to be a general backbone network.

2. Related Work

CNNs with Global Cues. In the deep learning era, CNNs can be regarded as a hierarchical ensemble of local features with different reception fields. Unfortunately, most CNNs [28, 36, 18, 38, 47, 22, 43] are good at extracting local features but experience difficulty to capture global cues.

To alleviate such a limitation, one solution is to define larger receptive fields by introducing deeper architectures and/or more pooling operations [21, 20]. The dilated convolution methods [48, 49] increased the sampling step size, while deformable convolution [13] learned the sampling positions. SENet [21] and GENet [20] proposed to use global Avgpooling to aggregate global context and then used it to reweight feature channels, while CBAM [45] respectively used global Maxpooling and global Avgpooling to refine features independently in the spatial and channel dimensions.

The other solution is the global attention mechanism [44, 7, 4, 19, 37], which has demonstrated great advantage in capturing long-distance dependencies in natural language processing [42, 15, 5]. Inspired by the non-local means method [6], the non-local operation [44] was introduced to CNNs in a self-attention manner so that the response at each position is a weighted sum of the features at all (global) positions. Attention augmented convolutional networks [4] concatenated convolutional feature maps with self-attentional feature maps to augment convolution operations for capturing long-range interactions. Relation Networks [19] proposed an object attention module, which processes a set of objects simultaneously through interaction between their appearance feature and geometry.

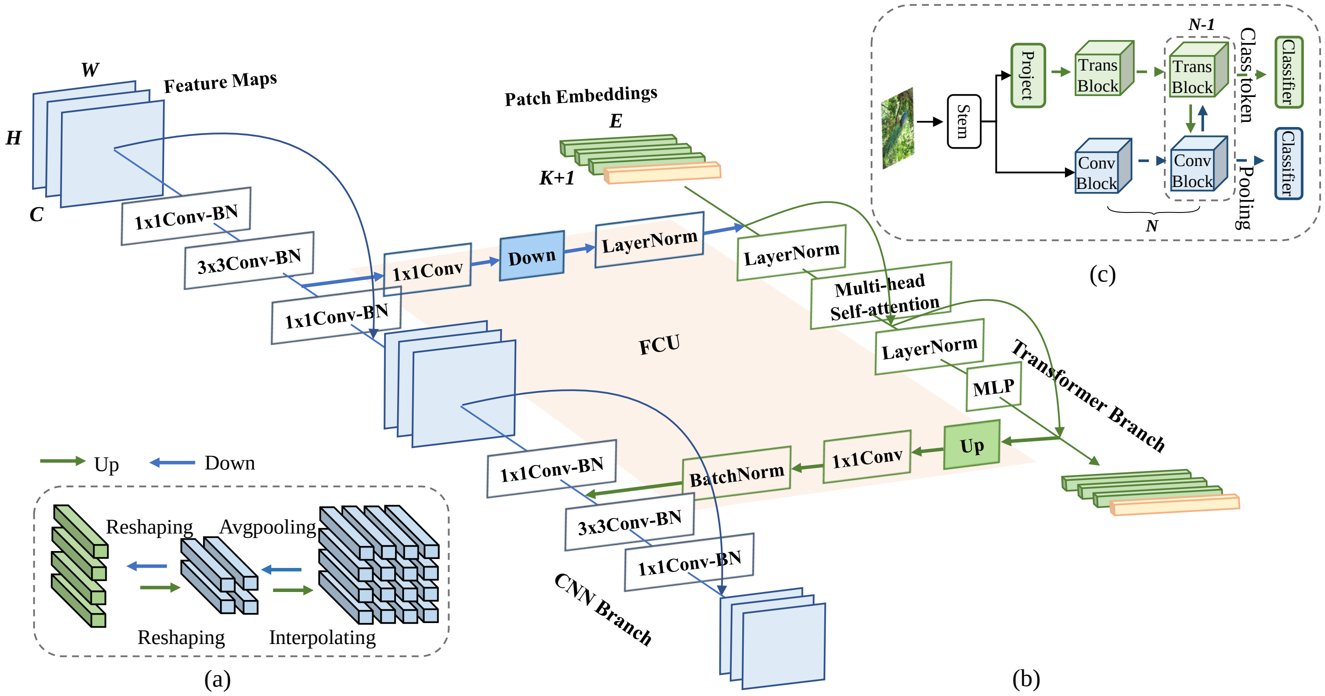


Figure 2: Network architecture of the proposed Conformer. (a) Up-sampling and down-sampling for spatial alignment of feature maps and patch embeddings. (b) Implementation details of the CNN block, the transformer block, and the Feature Coupling Unit (FCU). (c) Thumbnail of Conformer.

Despite of the progress, existing solutions that introduce global cues to CNNs have obvious disadvantages. For the first solution, larger receptive fields require more intensive pooling operations, which implies lower spatial resolution. For the second solution, if convolutional operations are not properly fused with attention mechanisms, local feature details could deteriorate.

Visual Transformers. As a pioneered work, ViT [16] validated the feasibility of pure transformer architectures for computer vision tasks. To leverage the long-distance dependencies, transformer blocks acted as independent architectures or were introduced to CNNs for image classification [46, 41, 50], object detection [8, 56, 3], semantic segmentation [53], image enhancement [9] and image generation [11, 27]. However, the self-attention mechanism in visual transformers often ignores local feature details. To solve, DeiT [41] proposed using a distillation token to transfer CNN-based features to visual transformer while T2T-ViT [50] proposed using a tokenization module to recursively reorganize the image to tokens considering neighboring pixels. The DETR method [8, 56] fed local features extracted by CNN to the transformer encoder-decoder to model the global relationships between features in a serial fashion.

Different from existing works, Conformer defines the first concurrent network structure which fuses features in an interactive fashion. Such a structure not only naturally inherits the structure advantages of both CNN and transformers but also retains the representation capability of local features and global representations to the maximum extent.

3. Conformer

3.1. Overview

Local features and global representations are important counterparts, which have been extensively studied in the long history of visual descriptors. Local features and their descriptors [33, 26, 34], which are compact vector representations of local image neighborhoods, have been the building blocks of many computer vision algorithms. Global representations include, but not limited to, contour representations, shape descriptors, and object typologies at long-distance [31]. In the deep learning era, CNN collects local features in a hierarchical manner via convolutional operations and retains the local cues as feature maps. Visual transformer is believed to aggregate global representations among the compressed patch embeddings in a soft fashion by the cascaded self-attention modules.

In order to take advantage of local features and global representations, we design a concurrent network structure, as shown in Fig. 2(c), termed Conformer. Considering

stage	output	**CNN Branch**	**FCU**	**Transformer Branch**	
c1	112×112	7×7, 64, stride 2			
	56×56	3×3 max pooling, stride 2			
c2	56×56,197	[1×1, 64 3×3, 64 1×1, 256]	-	4×4, 384, stride 4 [MHSA-6, 384 1×1, 1536 1×1, 384]	×1
		[1×1, 64 3×3, 64 1×1, 256]	$[1 \times 1, 384] \longrightarrow$	[MHSA-6, 384 1×1, 1536 1×1, 384]	×3
		[1×1, 64 3×3, 64 1×1, 256]	$\longleftarrow [1 \times 1, 64]$		
c3	28×28,197	[1×1, 128 3×3, 128 1×1, 512]	$[1 \times 1, 384] \longrightarrow$	[MHSA-6, 384 1×1, 1536 1×1, 384]	×4
		[1×1, 128 3×3, 128 1×1, 512]	$\longleftarrow [1 \times 1, 128]$		
c4	14×14,197	[1×1, 256 3×3, 256 1×1, 1024]	$[1 \times 1, 384] \longrightarrow$	[MHSA-6, 384 1×1, 1536 1×1, 384]	×3
		[1×1, 256 3×3, 256 1×1, 1024]	$\longleftarrow [1 \times 1, 256]$		
c5	7×7,197	[1×1, 256 3×3, 256 1×1, 1024]	$[1 \times 1, 384] \longrightarrow$	[MHSA-6, 384 1×1, 1536 1×1, 384]	×1
		[1×1, 256 3×3, 256 1×1, 1024]	$\longleftarrow [1 \times 1, 256]$		
classifier	1×1, 1	global pooling	-	class token	
		1×1,1000	-	1×1,1000	
Parameters		**37.7 M**			
MACs		**10.6 G**			

Table 1: Architecture of Conformer-S, where MHSA-6 denotes the multi-head self-attention with heads 6 in transformer block and the fc layer is viewed as 1×1 convolution here. In FCU column, the arrows represent the flow of feature. And in output column, 56×56,197 respectively mean the size of feature map is 56×56 and the number of embedded patches is 197.

the complementarity of the two-style features, within Conformer, we consecutively feed the global context from the transformer branch to feature maps, to reinforce the global perception capability of the CNN branch. Similarly, local features from the CNN branch are progressively fed back to patch embeddings, to enrich the local details of the transformer branch. Such a process constitutes the interaction.

In special, Conformer is composed of a stem module, dual branches, FCUs to bridge dual branches, and two classifiers (a fc layer) for the dual branches. The stem module, which is a 7×7 convolution with stride 2 followed by a 3×3 max pooling with stride 2, is used to extract initial local features ($e.g.$, edge and texture information), which are then fed to the dual branches. The CNN branch and transformer branch are composed of N ($e.g.$, 12) repeated convolution and transformer blocks, respectively, as described in Tab. 1. Such a concurrent structure implies that CNN and transformer branch can respectively preserve the local features and global representations to the maximum extent. FCU is proposed as a bridge module to fuse local features in the CNN branch with global representations in the transformer branch, Fig. 2(b). FCU is applied from the second block because the initialized features of the two branches are the same. Along the branches, FCU progressively fuses feature maps and patch embeddings in an interactive fashion.

Finally, for the CNN branch, all the features are pooled and fed to one classifier. For the transformer branch, the class token is taken out and fed to the other classifier. During training, we use two cross entropy losses to separately supervise the two classifiers. The importance of the loss functions are empirically set to be same. During inference, the outputs of the two classifiers are simply summarized as the prediction results.

3.2. Network Structure

CNN Branch. As shown in Fig. 2(b), the CNN branch adopts feature pyramid structure, where the resolution of feature maps decreases with network depth while the channel number increases. We split the whole branch into 4 stages, as described in Tab. 1(CNN Branch). Each stage is composed of multiple convolution blocks and each convolution block contains n_c bottlenecks. Following the definition in ResNet [18], a bottleneck contains a 1×1 down-projection convolution, a 3×3 spatial convolution, a 1×1 up-projection convolution, and a residual connection between the input and output of the bottleneck. In experiments, n_c is set to be 1 in the first convolution block and satisfies ≥ 2 in the subsequent $N - 1$ convolution blocks.

Visual transformers [16, 41] project an image patch into a vector through a single step, causing the lost of local details. While in CNNs, convolution kernels slide over feature maps with overlap, which provides the possibility to preserve fine-detailed local features. Consequently, the CNN branch is able to consecutively provide local feature details for the transformer branch.

Transformer Branch. Following ViT [16], this branch contains N repeated transformer blocks. As shown in Fig. 2(b), each transformer block consists of a multi-head self-attention module and an MLP block (contains a up-projection fc layer and a down-projection fc layer). LayerNorms [2] are applied before each layer and residual connections in both the self-attention layer and MLP block. For tokenization, we compress the feature maps generated by the stem module into 14×14 patch embeddings without

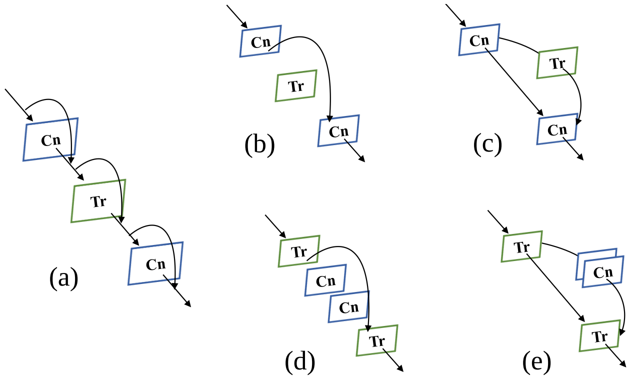
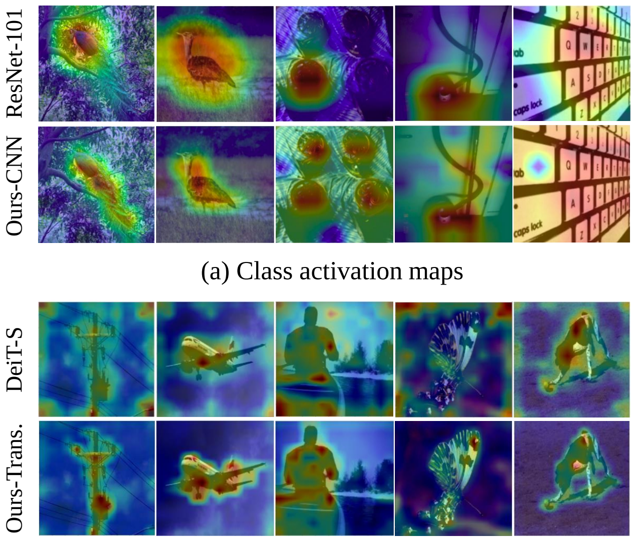


Figure 3: Structure analysis. C_n and T_r respectively denote a bottleneck and a transformer block. (a) The dual structure can be considered as a special serial case of the residual structure. (b) The CNN ($e.g.$, ResNet); (c) A special hybrid structure where the transformer block is embedded to bottlenecks. (d) The visual transformers ($e.g.$, ViT); (e) A special case where the bottlenecks are embedded to the transformer blocks.

overlap, by a linear projection layer, which is a 4×4 convolution with stride 4. A class token is then pretended to the patch embeddings for classification. Considering that the CNN branch (3×3 convolution) encodes both local features and spatial location information [ ], the positional embeddings are no longer required. This facilities increasing image resolution for downstream vision tasks.

Feature Coupling Unit. Given the feature maps in the CNN branch and patch embeddings in the transformer branch, how to eliminate the misalignment between them is an important issue. To solve, we propose the FCU to consecutively couple local features with global representations in an interactive manner.

On the one hand, we must realize that the feature dimensinalities of CNN and transformer are inconsistent. The CNN feature maps have the dimensinality $C \times H \times W$ (C, H, W are channels, height and width respectively), while the shape of the patch embeddings is $(K + 1) \times E$, where K, 1, and E respectively represent the number of image patches, class token and embedding dimensions. When fed to the transformer branch, feature maps first require to get through 1×1 convolution to align the channel numbers of the patch embeddings. A down-sampling module (Fig. 2(a)) is then used to complete the spatial dimension alignment. Finally, the feature maps are added with patch embeddings, as shown in Fig. 2(b). When fed back from the transformer branch to the CNN branch, the patch embeddings require to be up-sampled (Fig. 2(a)) to align the spatial scale. The channel dimension is then aligned with that of CNN feature maps through the 1×1 convolution, and added to the feature maps. Meanwhile, LayerNorm and BatchNorm modules are used to regularize features.



(a) Class activation maps

(b) Attention maps

Figure 4: Feature analysis. (a) Class activation maps in ResNet-101 and the CNN branch of Conformer-S by using the CAM method [55]. (b) Attention maps in DeiT-S and the transformer branch of Conformer-S by using the Attention Rollout method [1]. (Best viewed in color)

On the other hand, there is a significant semantic gap between feature maps and patch embeddings, $i.e.$, feature maps are collected from the local convolutional operators while patch embeddings are aggregated with the global self-attention mechanisms. FCU is therefore applied in each block (except the first) to progressively fill the semantic gap.

3.3. Analysis and Discussion

Structure Analysis. By considering the FCU as a short connection, we can abstract the proposed dual structure into the special serial residual structure, as shown in Fig. 3(a). Under different residual connection units, Conformer can implement different depths combinations of bottlenecks (as in ResNet, Fig. 3(b)) and transformer blocks (as in ViT, Fig. 3(d)), implying that Conformer inherits the structural advantages of both CNNs and visual transformers. Furthermore, it achieves different permutations of bottlenecks and transformer blocks at different depths, including but not limited to Figs. 3(c) and (e). This greatly enhances the representation capacity of the network.

Feature Analysis. We visualize the feature maps in Fig. 1, class activation maps and attention maps in Fig. 4. Compared with ResNet [18], with the coupled global representations, the CNN branch of Conformer tends to activate

Model	Image size	#Params (M)	MACs (G)	Top-1 (%)
ResNet-50 [18]	224^2	25.6	4.1	76.2
ResNet-101 [18]	224^2	44.5	7.8	77.4
ResNet-152 [18]	224^2	60.2	11.6	78.3
RegNetY-4.0GF [35]	224^2	20.6	4.0	78.8
RegNetY-12.0GF [35]	224^2	51.8	12.1	80.3
RegNetY-32.0GF [35]	224^2	145.0	32.3	81.0
ViT-B [16]	384^2	86	55.5	77.9
ViT-L [16]	384^2	307	191.1	76.5
T2T-ViT$_t$-14 [50]	224^2	21.5	5.2	80.7
T2T-ViT$_t$-19 [50]	224^2	39.0	8.4	81.4
T2T-ViT$_t$-24 [50]	224^2	64.1	13.2	82.2
DeiT-S [41]	224^2	22.1	4.6	79.8
DeiT-B [41]	224^2	86.6	17.6	81.8
Conformer-Ti	224^2	23.5	5.2	81.3
Conformer-S	224^2	37.7	10.6	83.4
Conformer-B	224^2	83.3	23.3	84.1

Table 2: Top-1 accuracy for image classification on the ImageNet validation set.

larger regions rather than local areas, suggesting enhanced long-distance feature dependencies, which are significantly demonstrated in Figs. 1(f) and 4(a). Thanks to the fine-detailed local features progressively provided by the CNN branch, the patch embeddings of the transformer branch in the Conformer retain important detailed local features (Figs. 1(d) and (h)), which are deteriorated by the visual transformers [16, 41] (Figs. 1(c) and (g)). Furthermore, the attention area in Fig. 4(b) is more complete while the background is significantly suppressed, implying the higher discriminative capacity of the learned feature representations by Conformer.

4. Experiments

4.1. Model Variants

By tuning the parameters of the CNN and transformer branches, we have the model variants, termed Conformer-Ti, -S, and -B, respectively. The details of Conformer-S are described in Tab. 1, and those of Conformer-Ti/B are in the Appendix. Conformer-S/32 splits the feature maps to 7×7 patches, *i.e.*, the patch size is 32×32 in the transformer branch.

4.2. Image Classification

Experimental Setting. Conformer is trained on the ImageNet-1k [14] training set with 1.3M images and tested upon the validation set. The Top-1 accuracy is reported in Tab. 2. To make the transformer converge to a reasonable performance, we follow the data augmentation and regularization techniques in DeiT [41]. These techniques include Mixup [52], CutMix [51], Erasing [54], Rand-Augment [12] and Stochastic Depth [23]). The model is trained for 300 epochs with the AdamW optimizer [32], batchsize 1024 and weight decay 0.05. The initial learning rate is set to 0.001 and decay in a cosine schedule.

Performance. Under similar parameters and computational budgets, Tab. 2, Conformers outperform both CNN and visual transformers. For example, Conformer-S (with 37.7M parameters and 10.6G MACs) respectively outperforms ResNet-152 (with 60.2M parameters and 11.6G MACs) by 4.1%((83.4% vs. 78.3%) and DeiT-B (with 86.6M parameters and 17.6G MACs) by 1.6% (83.4% vs. 81.8%). Conformer-B, with comparable parameters and moderate MAC cost, outperforms DeiT-B by 2.3% (84.1% vs. 81.8%). Beyond its superior performance, Conformer converges faster than the visual transformers.

4.3. Object Detection and Instance Segmentation

To verify Conformer's versatility, we test it on instance-level tasks (*e.g.*, object detection) and pixel-level tasks (*e.g.*, instance segmentation) on the MSCOCO dataset[1] [30]. Conformer, as the backbone, is migrated without extra design, and the relative accuracy and parameter comparison is included in Tab. 2. With the CNN branch, we can use the output feature maps of $[c_2, c_3, c_4, c_5]$ as side-output to construct the feature pyramid [29].

Experimental Setting. As is common practice, the models are trained on the MSCOCO training set and tested on the MSCOCO minival set. In Tab. 3, we report $\mathrm{AP}^{\mathrm{bbox}}$ ($\mathrm{AP}^{\mathrm{segm}}$), $\mathrm{AP}_S^{\mathrm{bbox}}$ ($\mathrm{AP}_S^{\mathrm{segm}}$), $\mathrm{AP}_M^{\mathrm{bbox}}$ ($\mathrm{AP}_M^{\mathrm{segm}}$), and $\mathrm{AP}_L^{\mathrm{bbox}}$ ($\mathrm{AP}_L^{\mathrm{segm}}$) for averaged over IoU thresholds, small, medium and large objects of box (mask), respectively. Unless explicitly specified, we use the batch size 32, with a learning rate 0.0002, optimizer AdamW [32], weight decay 0.0001 and max epoch 12. The learning rate decays at the 8-th and 11-th epoch by a magnitude.

Performance. As shown in Tab. 3, Conformer significantly boosts the $\mathrm{AP}^{\mathrm{bbox}}$ and $\mathrm{AP}^{\mathrm{segm}}$. For object detection, the mAP of Conformer-S/32 (55.4 M & 288.4 GFLOPs) is 3.7% higher than that of the FPN baseline (ResNet-101, 60.5 M & 295.7 GFLOPs). For instance segmentation, the mAP of Conformer-S/32 (58.1M & 341.4 GFLOPs) is 3.6% higher than that of the Mask R-CNN baseline (ResNet-101, 63.2 M & 348.8 GFLOPs). This demonstrates the importance of global representations for high level tasks and sug-

[1] Using mmdetection library at github.com/open-mmlab/mmdetection

Method	Backbone	Input size	#Params	GFLOPs	$\mathrm{AP^{bbox}}$	$\mathrm{AP^{bbox}_S}$	AP^{bbox}_M	$\mathrm{AP^{bbox}_L}$	$\mathrm{AP^{segm}}$	$\mathrm{AP^{segm}_S}$	$\mathrm{AP^{segm}_M}$	$\mathrm{AP^{segm}_L}$
FPN	ResNet-50† [29]	(1333, 800)	41.5 M	215.8	37.4	21.2	41.0	48.1	-	-	-	-
	ResNet-101† [29]	(1333, 800)	60.5 M	295.7	39.4	22.4	43.7	51.1	-	-	-	-
	Conformer-S/32	(1344, 800)	55.4 M	288.4	43.1	26.8	46.5	55.8	-	-	-	-
	Conformer-S/16	(1120, 800)	54.2 M	404.6	44.2	28.5	48.1	58.4	-	-	-	-
Mask R-CNN	ResNet-50† [17]	(1333, 800)	44.2 M	268.9	38.2	21.9	40.9	49.5	34.7	18.3	37.4	47.2
	ResNet-101† [17]	(1333, 800)	63.2 M	348.8	40.0	22.6	44.0	52.6	36.1	18.8	39.7	49.5
	Conformer-S/32	(1344, 800)	58.1 M	341.4	43.6	27.5	46.9	56.5	39.7	23.5	42.8	53.2
	Conformer-S/16	(1120, 800)	56.9 M	457.7	44.9	28.7	48.8	58.6	40.7	24.4	44.3	55.1

Table 3: Performance for object detection and instance segmentation on the MSCOCO minival set. † means the results are reported by the mmdetection library [10].

Transformer branch			CNN branch			p_p	MACs	Acc.(%)
E	d_h	#Params	n_c	C	#Params			
384	6	22 M	-	-	-	-	4.6 G	79.8
			2	64	1.5 M	0.07	5.2 G	81.3
				128	4.5 M	0.2	6.4 G	82.3
				192	9.3 M	0.4	8.2 G	82.8
				256	15.7 M	0.7	10.6 G	83.4
				320	23.7 M	1.0	13.7 G	83.6
			4	192	15.8 M	0.7	10.9 G	83.3
			3	256	21.4 M	1.0	13.0G	83.5
576	9	48.9 M	-	-	-	-	10.0 G	79.0
			2	256	16.4 M	0.3	16.3 G	83.6
				384	36.4 M	0.7	23.3 G	84.1
768	12	86 M	-	-	-	-	17.6 G	81.8
			2	256	17.6 M	0.2	24.2 G	83.0

Table 4: Performance under different parameter proportions. E and d_h respectively denote the embedding dimensions and the head in the multi-head attention module in the transformer branch. C and n_c respectively represent the channels of c_2 and the bottleneck number within each convolution block in the CNN branch. p_p is the proportion of CNN (including stem and FCUs) and transformer branch parameters.

gests the great potential of Conformer to be a general backbone network.

4.4. Ablation Studies

Number of Parameters. The parameters of the proposed Conformer are combinations of the CNN and transformer branches. The parameter proportion of the two branches is a hyper-parameter to be experimentally determined. In Tab. 4, we evaluate performance of the two branches under different parameter settings. For the CNN branch, we tune the parameters of the CNN branch by changing the channels and the number of bottlenecks, which respectively control the width and depth of the CNN branch. For the transformer branch, we tune the parameters by changing the numbers of embedding dimensions and heads. From Tab. 4, one can see that the accuracy is improved by increasing either parameters of the CNN or the transformer branch. More CNN parameters bring greater improvement while the computational cost overhead is lower.

Model	#Params	MACs	Accuracy
DeiT-S/32	22.9 M	1.1 G	73.8%
ResNet-26d & DeiT-S	36.5 M	3.7 G	80.2%
ResNet-50d & DeiT-S	**46.0 M**	5.5 G	80.4%
Conformer-S/32	38.8 M	**7.0 G**	**81.9%**

Table 5: Comparison of hybrid structures. DeiT-S/32 means the patch size is 32×32 for the DeiT-S model [41]. ResNet-26/50d is the variant of ResNet-26/50, and its stem module is composed of three 3×3 convolutions.

Dual Structure. Conformer is a dual model, which is totally different from the serial hybrid ViT (CNN → Transformer) [16]. In Tab. 5, ResNet-26/50d & DeiT-S is a hybrid model which consists of ResNet-26/50d [18] and DeiT-S [41], where DeiT-S forms tokens upon the feature maps extracted by ResNet-26/50d. With comparable computational cost overhead, Conformer-S/32 outperforms the serial hybrid model although ResNet-26/50d can retain more local information within the stem stage.

Positional Embeddings. Considering that the CNN branch (3×3 convolution) encodes both local features and spatial location information, the positional embeddings are assumed no longer required for Conformer. In Tab. 6, when the positional embedding is removed, the accuracy of DeiT-S decreases 2.4%, while that of Conformer-S decreases marginally (0.1%).

Sampling Strategies. In FCU, to make CNN-based feature maps coupling with Transformer-based patch

Method	Pos. Embeds	Accuracy
Deit-S	√	79.8%
	×	77.4% (**-2.4%**)
Conformer-S	√	83.5%
	×	83.4% (**-0.1%**)

Table 6: Comparison of positional embeddings strategies. "Pos. Embeds" is the abbreviation for "learnable positional embeddings".

Down	Up	#Params	MACs	Accuracy
Maxpooling	Interpolation	37.7 M	10.3 G	83.3%
Avgpooling	Interpolation	37.7 M	10.3 G	**83.4%**
Convolution	Interpolation	**47.7 M**	**12.3 G**	**83.4%**
Attention	Attention	39.4 M	11.3 G	83.3%

Table 7: Comparison of sampling strategies. The nearest neighbor interpolation is used.

Model	#Params	MACs	Acc^{Cn}	Acc^{Tr}	Acc^{All}
DeiT-S	22.0 M	4.2 G	-	79.8%	79.8%
ResNet-101	44.5 M	7.8 G	80.6%	-	80.6%
DeiT-S + ResNet-101	**66.5 M**	**11.2 G**	80.6%	79.8%	81.8%
Conformer-S	37.7 M	10.3 G	**83.3%**	**83.1%**	**83.4%**

Table 8: Performance comparison of ensemble models. Acc^{Cn} and Acc^{Tr} respectively denote the accuracy of the CNN and transformer branches.

embeddings, up/down-sampling operations are used to align the spatial scale. In Tab. 7, we compare different up/down-sampling strategies including Maxpooling, Avgpooling, convolution and attention-based sampling[2]. Compared with Max/Avgpooling sampling, convolution and attention-based sampling methods use more parameters and computation cost but achieve comparable accuracy. We thereby choose the Avgpooling strategy.

Comparison with Ensemble Models. Conformer is compared with the ensemble models combining the outputs of CNN and transformer. For fair comparison, we use the same data augmentation and regularization strategies and the same training epochs (300) to train ResNet-101 [ ], and combine it with the DeiT-S [ ] model to form an ensemble model, and report the accuracy in Tab. 8. The accuracies of the CNN branch, the transformer branch, and the Conformer-S respectively reach 83.3%, 83.1%, and 83.4%. In contrast, the ensemble model (DeiT-S+ResNet-101) archives 81.8%, which is 1.6% lower than that of Conformer-S (83.4%), although it uses significantly more parameters and MACs.

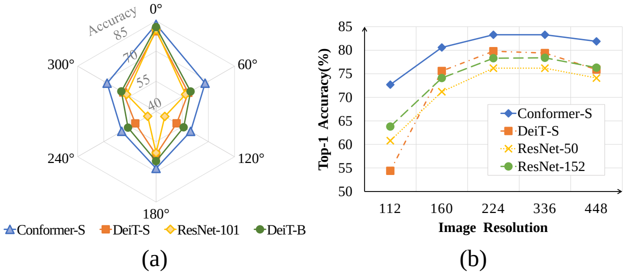


Figure 5: Generalization capability. (a) Comparison of rotation invariance. The compared models are trained under the same data augmentation settings and directly evaluated on rotated images without model fintuning. (b) Comparison of scale invariance. The models are trained on images with the resolution of 224×224, and tested on different image resolutions without model finetuning.

4.5. Generalization Capability

Rotation Invariance. To verify the generalization capability of the model in terms of rotation, we rotate test images by 0°, 60°, 120°, 180°, 240° and 300° and evaluate the performance of models trained under same data augmentation settings. As shown in Fig. 5(a), all models report comparable performance for images without rotation (0°). For the rotated test images, the performance of ResNet-101 drops significantly. In contrast, Conformer-S reports higher performance, which implies stronger rotation invariance.

Scale Invariance. In Fig. 5(b), we compare the scale adaptation ability of Conformer with those of visual transformers (DeiT-S) and CNN (ResNet). We interpolate the positional embeddings of DeiT-S to adapt it to input images of different resolutions during inference. When the size of input images reduces from 224 to 112, DeiT-S's performance drops by 25% and that of ResNet-50/152 drops by 15%. In contrast, the performance of Conformer drops only by 10%, demonstrating higher scale invariance of the learned feature representations.

5. Conclusion

We propose Conformer, the first dual backbone to combining CNN with visual transformer. Within Conformer, we leverage the convolution operators to extract local features and the self-attention mechanisms to capture global representations. We design the Feature Coupling Unit (FCU) to fuse local features and global representations, enhancing the ability of visual representations in an interactive fashion.

[2] Refer to Appendix for detailed attention-based sampling.

Experiments show that Conformer, with comparable parameters and computation budgets, outperforms both conventional CNNs and visual transformers, in striking contrast with the state-of-the-arts. On downstream tasks, Conformer has shown the great potential to be a simple yet effective backbone network.

References

[1] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*, 2020. 5

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 2, 4

[3] Josh Beal, Eric Kim, Eric Tzeng, Dong Huk Park, Andrew Zhai, and Dmitry Kislyuk. Toward transformer-based object detection. *arXiv preprint arXiv:2012.09958*, 2020. 2, 3

[4] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V. Le. Attention augmented convolutional networks. In *IEEE ICCV*, pages 3286–3295, 2019. 2

[5] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. 2

[6] Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. A non-local algorithm for image denoising. In *IEEE CVPR*, pages 60–65, 2005. 2

[7] Yue Cao, Jiarui Xu, Stephen Lin, Fangyun Wei, and Han Hu. Gcnet: Non-local networks meet squeeze-excitation networks and beyond. In *IEEE CVPRW*, 2019. 2

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 2, 3

[9] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. *arXiv preprint arXiv:2012.00364*, 2020. 2, 3

[10] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. 7

[11] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *ICML*, pages 1691–1703. PMLR, 2020. 3

[12] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *IEEE CVPRW*, pages 702–703, 2020. 6

[13] Jifeng Dai, Haozhi Qi, Yuwen Xiong, Yi Li, Guodong Zhang, Han Hu, and Yichen Wei. Deformable convolutional networks. In *IEEE ICCV*, pages 764–773, 2017. 2

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE CVPR*, pages 248–255, 2009. 6

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 2

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 3, 4, 6, 7

[17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and B. Ross Girshick. Mask r-cnn. In *IEEE ICCV*, pages 386–397, 2017. 7

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE CVPR*, pages 770–778, 2016. 1, 2, 4, 5, 6, 7, 8, 11

[19] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *IEEE CVPR*, pages 3588–3597, 2018. 2

[20] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Andrea Vedaldi. Gather-excite: Exploiting feature context in convolutional neural networks. *arXiv preprint arXiv:1810.12348*, 2018. 2

[21] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *IEEE CVPR*, pages 7132–7141, 2018. 1, 2

[22] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *IEEE CVPR*, pages 4700–4708, 2017. 2

[23] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *ECCV*, pages 646–661. Springer, 2016. 6

[24] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, pages 448–456. PMLR, 2015. 2

[25] Md Amirul Islam, Matthew Kowal, Sen Jia, Konstantinos G Derpanis, and Neil DB Bruce. Position, padding and predictions: A deeper look at position information in cnns. *arXiv preprint arXiv:2101.12322*, 2021. 5

[26] Anil K. Jain and Farshid Farrokhnia. Unsupervised texture segmentation using gabor filters. *Pattern Recognit.*, 24(12):1167–1186, 1991. 3

[27] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. Transgan: Two transformers can make one strong gan. *arXiv preprint arXiv:2102.07074*, 2021. 2, 3

[28] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NeuralIPS*, volume 25, pages 1097–1105, 2012. 1, 2

[29] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *IEEE CVPR*, pages 936–944, 2017. 6, 7

[30] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 6

[31] Dimitri A. Lisin, Marwan A. Mattar, Matthew B. Blaschko, Erik G. Learned-Miller, and Mark C. Benfield. Combining local and global image features for object class recognition. In *IEEE CVPRW*, pages 47–55, 2008. 3

[32] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[33] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.*, 60(2):91–110, 2004. 3

[34] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(7):971–987, 2002. 3

[35] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. *arXiv preprint arXiv:2003.13678*, 2020. 6, 11

[36] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 2

[37] Aravind Srinivas, Tsung-Yi Lin, Niki Parmar, Jonathon Shlens, Pieter Abbeel, and Ashish Vaswani. Bottleneck transformers for visual recognition. *arXiv preprint arXiv:2101.11605*, 2021. 2

[38] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 31, 2017. 2

[39] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *IEEE CVPR*, pages 1–9, 2015. 1

[40] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *ICML*, pages 6105–6114. PMLR, 2019. 11

[41] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020. 1, 2, 3, 4, 6, 7, 8, 11

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 2

[43] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, et al. Deep high-resolution representation learning for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2020. 2

[44] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *IEEE CVPR*, pages 7794–7803, 2018. 2

[45] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *ECCV*, pages 3–19. Springer, 2018. 2

[46] Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Masayoshi Tomizuka, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision. *arXiv preprint arXiv:2006.03677*, 2020. 2, 3

[47] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *IEEE CVPR*, pages 1492–1500, 2017. 1, 2

[48] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 2

[49] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *IEEE CVPR*, pages 472–480, 2017. 2

[50] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986*, 2021. 2, 3, 6

[51] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *IEEE ICCV*, pages 6023–6032, 2019. 6

[52] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017. 6

[53] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. *arXiv preprint arXiv:2012.15840*, 2020. 2, 3

[54] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *AAAI*, volume 34, pages 13001–13008, 2020. 6

[55] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *IEEE CVPR*, pages 2921–2929, 2016. 5

[56] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020. 3

A. Model Architectures

The architectures of Conformer-Ti/B are detailed in Tab. 12. Compared with Conformer-S, Conformer-Ti reduces channel number of the CNN branch by 1/4, and Conformer-B increases channel number in the CNN branch, head number of the multi-head attention module and the embedding dimensions in the transformer branch by 1.5.

B. Attention-based Sampling

We also design a down-sampling-up-sampling strategy based on the cross attention between feature maps and patch embeddings.

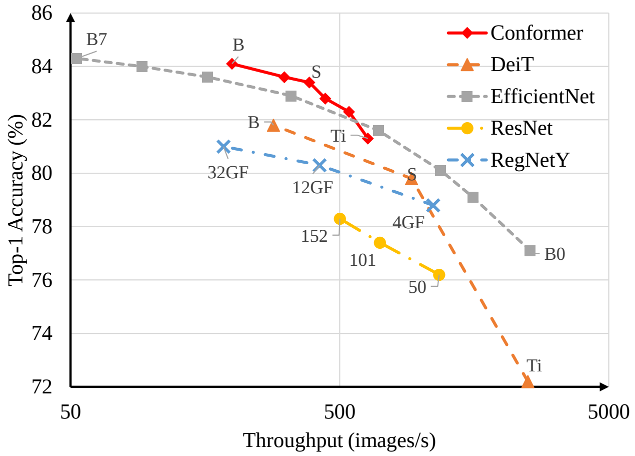


Figure 6: Throughput and accuracy on ImageNet of Conformer compared to DeiT [41], ResNet [18], RegNetY [35] and EfficientNet [40]. The throughput is measured as the number of images processed per second on a 32GB V100 GPU.

Let h, w and c respectively denote the height, width, channel of feature maps in a block (we omit the batch dimension here for simplicity), K and E respectively represent the number of patch embeddings (termed P_t) and channel dimension in the transformer branch. We split the feature maps into K patches (*e.g.*, 14×14), termed P_c. The dimension of each patch is $n \times c$. After aligning the channel dimension by 1×1 convolution, the shape of each patch is $n \times E$.

For down sampling, the fusion between patch i in P_c (denoted P_c^i) and patch j in P_t (denoted P_t^j) is formulated as

$$P_t^j = P_t^j + \text{Softmax}\left(\frac{(P_t^j W_q)(P_c^i W_k)^T}{\sqrt{E}}\right)(P_c^i W_v), \quad (1)$$

where $W_q, W_k, W_v \in \mathbb{R}^{E \times E}$ are learned linear transformations which map the input P_t^j to queries Q, keys K and values V, respectively.

For up sampling, we re-use the attention weights in Eq. 1 and formulate the process as

$$\tilde{P}_c^i = \tilde{P}_c^i + \text{Softmax}\left(\frac{(P_t^j W_q)(P_c^i W_k)^T}{\sqrt{E}}\right)^T \tilde{P}_t^j, \quad (2)$$

where $\tilde{P}_c^i$ and $\tilde{P}_c^i$ respectively denote that P_c^i is processed by convolution layers and P_t^j by a transformer block (Fig. 2 in the paper).

C. Inference Time

Classification. Following DeiT [41], we evaluate and compare the throughput of various methods in Fig. 6. One can see that our Conformer outperforms EfficientNet [40] under comparable throughput.

Method	Backbone	#Params (M)	GLOPs	FPS
FPN	ResNet-50	41.5	215.8	20.2
	ResNet-101	60.5	295.7	15.9
	Conformer-S/32	55.4	288.4	13.5
	Conformer-S	54.2	404.6	8.2
Mask R-CNN	ResNet-50	44.2	268.9	13.2
	ResNet-101	63.2	348.8	11.5
	Conformer-S/32	58.1	341.4	10.9
	Conformer-S	56.9	457.7	7.1

Table 9: Comparison of inference time. FPS is measured on a 32GB V100 GPU with batchsize 1.

Object detection and instance segmentation. Similarly, we measure Frame Per Second (FPS) as the inference speed and show the comparison in the Tab. 9. Combining Tab.3 in the paper and Tab. 9 here, compared with ResNet-101 [18], Conformer-S/32 has the comparable parameters, GFLOPs and inference speed, but can outperform ResNet-101 by a significant margin on both object detection and instance segmentation tasks, which further demonstrates the potential to be a general backbone network.

D. Residual Structure

As shown in Fig. 3 in the paper, by considering FCUs as short connection we abstract Conformer with a dual structure to a serial structure with residual connections. In other words, under different residual connections, Conformer can degenerate to different sub-structures. We test some sub-structures and report the corresponding performance in Tab. 10. From Tab. 10, one can see that the proposed residual structure outperforms other sub-structures.

Index	#Params (M)	MACs (G)	Accuracy (%)
1	8.6	9.2	73.9
2	37.0	10.8	80.8
3	22.1	4.6	79.8
4	28.9	6.0	80.2
Conformer-S	37.7	10.6	83.4

Table 10: Performance of Conformer sub-structures. Where the index 1, 2, 3 and 4 respectively represent the substructures shown in Figs. 3(b), (c), (d) and (e).

E. Fusion Interval

In the paper, we proposed a Feature Coupling Unit to interact the local features and global representations in each block to progressively align the features to fill the semantic gap. To validate whether fusion should be done in each block, we conduct experiments on fusion intervals and report the performance on ImageNet in Tab. 11. From Tab. 11, one can see that smaller fusion intervals report higher performance, implying that frequent interaction facilities the representation learning.

Interval	#Params (M)	MACs (G)	Accuracy (%)
1	37.7	10.6	83.4
2	34.2	9.2	82.9
4	32.3	8.4	82.2

Table 11: Comparison of fusion intervals. 1, 2 and 4 respectively represent performing fusion every 1, 2 and 4 block(s).

F. Convergence speed

For the convolution operations introduced, Fig. 7, both the CNN branch and the transformer branch of Conformer-S significantly outperforms DeiT during the first 50 epochs. This demonstrates the inductive bias of convolution facilities the convergence of visual transformers.

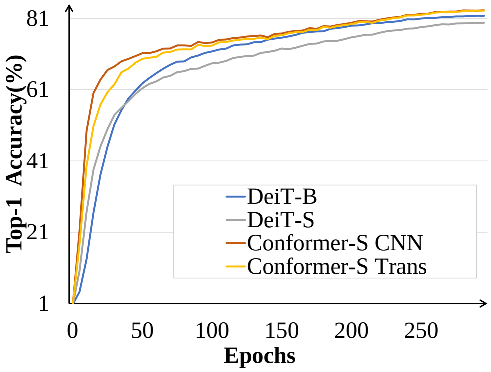


Figure 7: Training Accuracy on the val set.

	CNNTransformer-Ti					**CNNTransformer-B**				
stage	output	**CNN Branch**	**fcuc**	**Transformer Branch**		output	**CNN Branch**	**fcuc**	**Transformer Branch**	
c1	112×112	7×7, 64, stride 2				112×112	7×7, 64, stride 2			
	56×56	3×3 max pooling, stride 2				56×56	3×3 max pooling, stride 2			
c2	56 × 56,197	$\begin{bmatrix} 1\times1, 16 \\ 3\times3, 16 \\ 1\times1, 64 \end{bmatrix}$	-	4×4, 384, stride 4 $\begin{bmatrix} \text{MHSA-6, 384} \\ 1\times1, 1536 \\ 1\times1, 384 \end{bmatrix}$	×1	56 × 56,197	$\begin{bmatrix} 1\times1, 96 \\ 3\times3, 96 \\ 1\times1, 384 \end{bmatrix}$	-	4×4, 384, stride 4 $\begin{bmatrix} \text{MHSA-9, 576} \\ 1\times1, 2304 \\ 1\times1, 576 \end{bmatrix}$	×1
		$\begin{bmatrix} 1\times1, 16 \\ 3\times3, 16 \\ 1\times1, 64 \end{bmatrix}$ - - - - - - - - - - - - $\begin{bmatrix} 1\times1, 16 \\ 3\times3, 16 \\ 1\times1, 64 \end{bmatrix}$	$[1 \times 1, 384] \longrightarrow$ $\longleftarrow [1 \times 1, 16]$	$\begin{bmatrix} \text{MHSA-6, 384} \\ 1\times1, 1536 \\ 1\times1, 384 \end{bmatrix}$	×3		$\begin{bmatrix} 1\times1, 96 \\ 3\times3, 96 \\ 1\times1, 384 \end{bmatrix}$ - - - - - - - - - - - - $\begin{bmatrix} 1\times1, 96 \\ 3\times3, 96 \\ 1\times1, 384 \end{bmatrix}$	$[1 \times 1, 576] \longrightarrow$ $\longleftarrow [1 \times 1, 96]$	$\begin{bmatrix} \text{MHSA-9, 576} \\ 1\times1, 2304 \\ 1\times1, 576 \end{bmatrix}$	×3
c3	28 × 28,197	$\begin{bmatrix} 1\times1, 32 \\ 3\times3, 32 \\ 1\times1, 128 \end{bmatrix}$ - - - - - - - - - - - - $\begin{bmatrix} 1\times1, 32 \\ 3\times3, 32 \\ 1\times1, 128 \end{bmatrix}$	$[1 \times 1, 384] \longrightarrow$ $\longleftarrow [1 \times 1, 32]$	$\begin{bmatrix} \text{MHSA-6, 384} \\ 1\times1, 1536 \\ 1\times1, 384 \end{bmatrix}$	×4	28 × 28,197	$\begin{bmatrix} 1\times1, 192 \\ 3\times3, 192 \\ 1\times1, 768 \end{bmatrix}$ - - - - - - - - - - - - $\begin{bmatrix} 1\times1, 192 \\ 3\times3, 192 \\ 1\times1, 768 \end{bmatrix}$	$[1 \times 1, 576] \longrightarrow$ $\longleftarrow [1 \times 1, 192]$	$\begin{bmatrix} \text{MHSA-9, 576} \\ 1\times1, 2304 \\ 1\times1, 576 \end{bmatrix}$	×4
c4	14 × 14,197	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}$ - - - - - - - - - - - - $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}$	$[1 \times 1, 384] \longrightarrow$ $\longleftarrow [1 \times 1, 64]$	$\begin{bmatrix} \text{MHSA-6, 384} \\ 1\times1, 1536 \\ 1\times1, 384 \end{bmatrix}$	×3	14 × 14,197	$\begin{bmatrix} 1\times1, 384 \\ 3\times3, 384 \\ 1\times1, 1536 \end{bmatrix}$ - - - - - - - - - - - - $\begin{bmatrix} 1\times1, 384 \\ 3\times3, 384 \\ 1\times1, 1536 \end{bmatrix}$	$[1 \times 1, 576] \longrightarrow$ $\longleftarrow [1 \times 1, 384]$	$\begin{bmatrix} \text{MHSA-9, 576} \\ 1\times1, 2304 \\ 1\times1, 576 \end{bmatrix}$	×3
c5	7 × 7,197	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}$ - - - - - - - - - - - - $\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}$	$[1 \times 1, 384] \longrightarrow$ $\longleftarrow [1 \times 1, 64]$	$\begin{bmatrix} \text{MHSA-6, 384} \\ 1\times1, 1536 \\ 1\times1, 384 \end{bmatrix}$	×1	7 × 7,197	$\begin{bmatrix} 1\times1, 384 \\ 3\times3, 384 \\ 1\times1, 1536 \end{bmatrix}$ - - - - - - - - - - - - $\begin{bmatrix} 1\times1, 384 \\ 3\times3, 384 \\ 1\times1, 1536 \end{bmatrix}$	$[1 \times 1, 576] \longrightarrow$ $\longleftarrow [1 \times 1, 384]$	$\begin{bmatrix} \text{MHSA-9, 576} \\ 1\times1, 2304 \\ 1\times1, 576 \end{bmatrix}$	×1
Parameters	**23.5 M**					**83.3 M**				
MACs	**5.2 G**					**23.3 G**				

Table 12: Architecture of CNNTransformer-Ti and CNNTransformer-B, where MHSA-6/9 denotes the multi-head self-attention with heads 6/9 in transformer block and the fc layer is viewed as 1×1 convolution here. And in output column, 56×56,197 respectively mean the size of feature map is 56×56 and the number of embedded patches is 197.