Pattern Recognition Letters 169 (2023) 95–101

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

FlexFormer: Flexible Transformer for efficient visual recognition☆

Xinyi Fan, Huajun Liu*

School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China

ARTICLE INFO

Article history:
Received 6 October 2022
Revised 22 February 2023
Accepted 31 March 2023
Available online 2 April 2023

Edited by: Prof. S. Sarkar

Keywords:
Vision transformer
Frequency analysis
Image classification

ABSTRACT

Vision Transformers have shown overwhelming superiority in computer vision communities compared with convolutional neural networks. Nevertheless, the understanding of multi-head self attentions, as the de facto ingredient of Transformers, is still limited, which leads to surging interest in explaining its core ideology. A notable theory interprets that, unlike high-frequency sensitive convolutions, self-attention behaves like a generalized spatial smoothing and blurs the high spatial-frequency signals with depth increasing. In this paper, we design a Conv-MSA structure to extract efficient local contextual information and remedy the inherent drawback of self-attention. Accordingly, a flexible transformer structure named ***FlexFormer***, with linear computational complexity on input image size, is proposed. Experimental results on several visual recognition benchmarks show that our FlexFormer achieved the state-of-the-art results on visual recognition tasks with fewer parameters and higher computational efficiency.

1. Introduction

Notable progress in deep learning nowadays is the introduction of Transformers [1,2] from natural language processing (NLP) to the computer vision (CV) community. In the past decades, many state-of-the-art image classification algorithms have been proposed primarily based on convolutional neural networks (CNNs) [3–5]. Very recently, vision Transformers [2,6,7] were introduced to computer vision community and excelled in many vision fields such as semantic segmentation [8], object detection [9], GAN [10], etc.

In essence, both convolutions and self-attentions (SAs, the core of Transformer) address the fundamental representation problem of structured data (e.g. image and video). Meriting from weak inductive bias and long-range dependencies compared to convolutions [2,11], vision Transformers can enhance generalization ability thus boosting performance on several tasks. Nevertheless, a fatal drawback of the SA mechanism is that excessive spatial smoothing in SAs leads to high-frequency signals restraining. More and more researches [12,13] have proved that self-attention mechanism is implicitly equivalent to a low-pass filter, which indicates that when ViT scales up its depth, fine-grained semantic information (such as edge, texture and lines etc.) involved in high-frequency signals is inhibited and only coarse-grained components are preserved. Recently, a series of researches have been conducted to counteract this problem. For instance, FcaNet [14] and NomMer [15] utilized discrete cosine transform (DCT) to select potentially useful frequency components and are certified to be more effective. Some previous works [6,16] have proved that local multi-head self attentions (MSAs), which restrict attention calculation in smaller patches, tended to achieve better results than global MSAs not only on large datasets like ImageNet [17] but also on small ones such as CIFAR [18]. Based on the success of CNNs on visual tasks, many former works tried to marry convolution with vision Transformer to improve performance. For instance, Conformer [19] and Mobile-Former [20] successfully assembled an independent convolutional branch with vision Transformer to fuse convolutional features and MSA representations. Noting that convolution operations could extract abundant contextual information, some other works embedding convolution operations to self-attention would like to introduce locality into Transformers. For instance, CPVT [21] and CvT [22] substituted the traditional absolute position encoding with convolutional dynamic position encoding and showed improvements in various tasks. However, many previous methods paid too much attention on transferring transformer structure from NLP to CV with structure optimization, complexity reduction and so on, and ignoring the transition smoothing property of self-attention, thus inevitably falling into collapse with depth increasing.

☆ Editor: Prof. S. Sarkar.

* Corresponding author.
E-mail addresses: xyxy@njust.edu.cn (X. Fan), liuhj@njust.edu.cn (H. Liu).

https://doi.org/10.1016/j.patrec.2023.03.028
0167-8655/

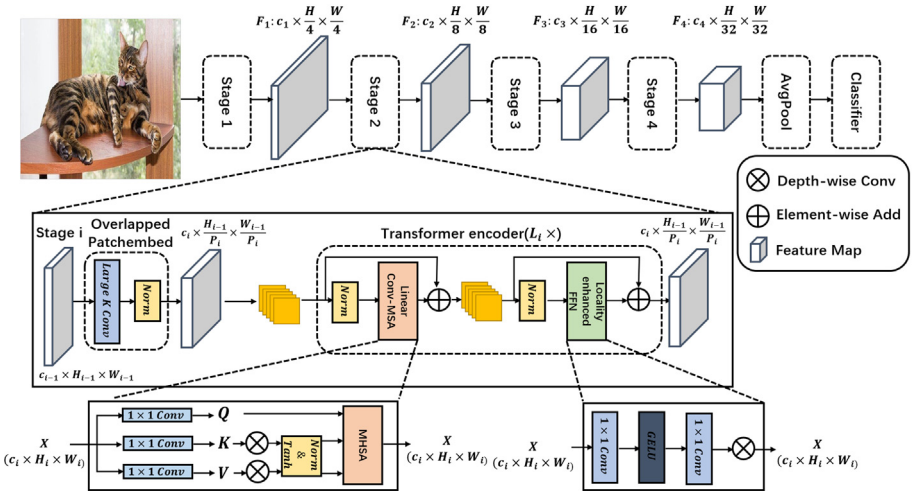


Fig. 1. Overall architecture of the proposed FlexFormer.

On this basis, we propose a Conv-MSA structure to remedy the limitation of MSAs. Moreover, based on the pyramid vision Transformer model [9], a four-stage multi-scale model, named FlexFormer, is proposed for visual recognition tasks. The overall architecture could be seen in Fig. 1.

The main contributions of this work are summarized as follows,

- Proposing an effective Conv-MSA structure based on contextual querying dot-products to improve the fine-grained features representation ability of self-attentions.
- Introducing a Tanh-Softmax hybrid nonlinearization method in a linear self-attention for fast convergence on visual recognition tasks.
- Proposing a FlexFormer model based on efficient attention and lightweight convolutions, which achieved the state-of-the-art recognition accuracy on several benchmarks with fewer parameters and higher computation efficiency.

2. Our approach

2.1. Revisiting self-attention

SAs [1,2], as the key portion of Transformer, compute the compatibility between the "query" (the token in consideration) and the "key" (the token being matched with) to weight the "value" (the token being matched with). The weights of SA are dynamically calculated based on the similarity between every pair of tokens, which provides a new modeling approach that is potentially more adaptive and general. Denoting that $\mathbf{Q}, \mathbf{K}, \mathbf{V} \in \mathbb{R}^{C\times H\times W}$ are in a broad sense *query*, *key*, and *value* matrices respectively, and are formally projected from the input of lth encoder block $\mathbf{Z}_l \in \mathbb{R}^{C\times H\times W}$, where (H, W) defines the spatial dimension and C defines the channel dimension, and C_h is a scaling factor, which is often set as the number of single-head channels, SA is defined as below,[1]

$$\mathrm{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}^{\mathrm{T}} \times \mathbf{K}}{\sqrt{C_h}}\right) \times \mathbf{V}^{\mathrm{T}} \tag{1}$$

[1] All non-bold letters represent scalars. Bold capital letter $\mathbf{X}$ denotes a matrix; Bold lower-case letters $\mathbf{x}$ is a column vector. $\mathbf{x}_i$ represents the ith column vector of the matrix $\mathbf{X}$. x_j denotes the jth element of $\mathbf{x}$. For convenience, we define that if $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$, $\mathbf{X}^{\mathrm{T}} \in \mathbb{R}^{H\times W\times C}$. $\mathbf{X} \times \mathbf{Y}$, $\mathbf{X} \cdot \mathbf{Y}$ and $\mathbf{X} + \mathbf{Y}$ denotes dot-product, element-wise product and element-wise add between two matrices respectively.

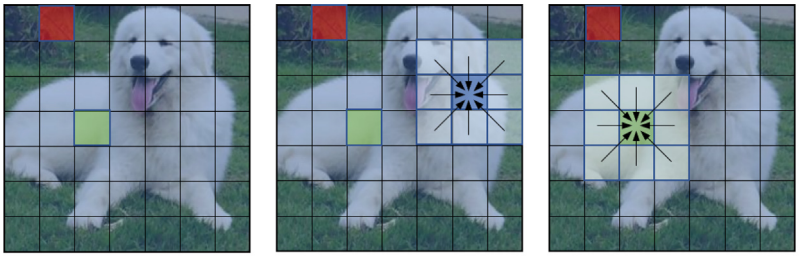
For more effective attention on different representation subspaces, MSA concatenates the output from several single-head attentions and projects it with another parameter matrix:

$$\mathrm{MSA}(\mathbf{Z}_l) = \mathrm{Concat}\left(\mathrm{head}_{1,l}, \ldots, \mathrm{head}_{H,l}\right) \tag{2}$$

2.2. Conv-MSA

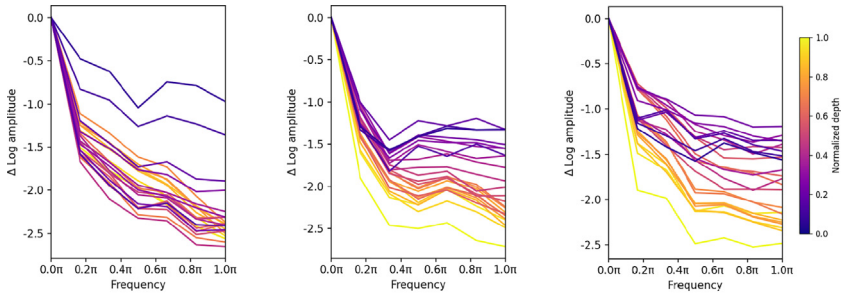
MSA is a transformation on all feature maps with large-sized and data-specific kernels in analogy with convolutions [12], thus they exhibit similar functions to some extent. Many previous works [23–26] have proved that replacing the convolution operation with self-attention is feasible. However, recent studies [12,13] gave a theoretical justification on self-attention that MSAs and Convs exhibit opposite behaviors. As a matter of fact, MSAs tend to blur the feature maps and reduce variance of features within the compatibility matrix. On the contrary, CNNs naturally extract fine-level high frequency details (e.g. edges, texture, lines) and implicitly encode absolute position information as a clue for decision making [27]. To be more specific, MSAs are similar to low-pass filters which suppress high-frequency signals and concentrate on global smoothed feature maps, while convolutions, conversely, act like high-pass filters and gradually refine the detailed feature maps. Therefore, MSAs and Convs are complementary. Some efforts have been made to combine Conv and MSA to improve the performance of Transformers. Typical examples including CvT [22] and CPVT [21] are illustrated in Fig. 2(b). Above both methods aim to extract dynamic position information by replacing absolute position encoding with their proposed conditional position encoding. There is no doubt that absolute positional encoding, a crucial component in the original vision Transformers, is less effective than convolutional dynamic position encoding. However, as can be seen from Fig. 3(b), high-frequency signals remain restrained and the representation ability of MSA has not been improved obviously in late stages.

According to the analysis above, we design convolution-embedded self-attention to counter the problem of the high-frequency signal suppression. Inspired by the former windows Transformer [6,28], we hope that each query token's receptive field could be expanded to a larger neighboring region around its corresponding tokens in the key-value pair. Furthermore, we centralize the contextual information around one token with convolution operations to preserve high-frequency details, as shown in Fig. 2(c). Finally, by two cascaded contextual querying dot-products of linear



(a) Point query (ViT) (b) Dynamic query (CvT & CPVT) (c) Contextual query (Ours)

Fig. 2. Query comparison of different SA mechanism.



(a) For ViT (b) For CPVT (c) For ours

Fig. 3. Relative log amplitudes of Fourier transformed feature maps. Δ Log amplitude is the difference between the log amplitude at normalized frequency 0.0π (center) and 1.0π (boundary).

self-attentions, fine-grained signals should be preserved as much as possible in our Conv-MSA. Denoting the input $\mathbf{X} \in \mathbb{R}^{C\times H\times W}$ and *query*, *key* and *value* projected with 1×1 convolutions, depth-wise convolutions (DWConv) [29] are applied on *key* and *value* respectively to harvest the fine-grained features (i.e., edge, texture and lines etc). It could be formulated as:

$$\tilde{\mathbf{K}} = \mathrm{DWConv}(\mathbf{K}) \triangleq \sum_{p,q} \overline{\mathbf{K}}^{\mathbf{K}}_{p,q} \cdot \mathbf{K}^{k}_{i,j} \tag{3}$$

$$\tilde{\mathbf{V}} = \mathrm{DWConv}(\mathbf{V}) \triangleq \sum_{p,q} \overline{\mathbf{K}}^{\mathbf{V}}_{p,q} \cdot \mathbf{V}^{k}_{i,j} \tag{4}$$

$$\mathrm{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}^{\mathrm{T}} \times \tilde{\mathbf{K}}}{\sqrt{C_h}}\right) \times \tilde{\mathbf{V}}^{\mathrm{T}} \tag{5}$$

where $\overline{\mathbf{K}}^{\mathbf{F}}_{p,q} \in \mathbb{R}^{C}$, $p, q \in \{0, 1, \ldots, k-1\}$, represents the kernel weights with regard to the indices of the kernel position (p, q), and $\mathbf{V}^{k}_{i,j}$ and $\mathbf{K}^{k}_{i,j}$ refer to a $k \times k$ neighbourhood of location (i, j) in the tensor $\mathbf{K}$ and $\mathbf{V}$. To better explain its effectiveness, we compare the feature maps of original ViT, CPVT and ViT with Conv-MSA by fast Fourier Transformation (FFT) analysis. The relative log amplitudes of feature maps are listed in Fig. 3, where vertical axis refers to the amplitude of signal and horizontal axis refers to signal frequency from low (0.0π) to high (1.0π). And the color caption ranging from purple (0.0) to yellow (1.0) refers to the normalized depth. It can be seen that our Conv-MSA could better restrain the decreasing of amplitude of high-frequency signals compared with ViT and CPVT. Especially the attenuation of high frequency amplitude of ViT and CPVT (nearly -2.8) in the late stage is more serious than that of Conv-MSA (about -2.4). This could serve as an evidence that Conv-MSA is more capable of focusing on high-frequency signals. More evidence could be seen in Section 3.3.

2.3. FlexFormer

Based on the proposed Conv-MSA, we design our FlexFormer, an effective and efficient model that not only outperforms the existing self-attention module utilized in vision Transformers but also takes less parameters and computational cost. Details are shown in Fig. 4. The dot-product similarity is the core mechanism for long-range token interaction in Transformers, while excessive overhead on computation and storage is a challenging issue. Actually, in the original and most recent vision Transformer models, calculating the similarity of tokens as $\mathbf{Q}^{T} \otimes \mathbf{K}$ is based on position querying on a high-dimensional similarity matrix (e.g. $HW \times HW$), which leads to $\mathrm{O}((HW)^2)$ memory complexity and $\mathrm{O}((HW)^2C)$ computational complexity and builds a barrier for high-resolution images recognition and lightweight deployment on consumer electronic devices. On the contrary, calculating the similarity of tokens as $\mathbf{Q} \otimes \mathbf{K}^{T}$ can obtain a $C \times C$ similarity matrix, and querying channel features on this lower-dimensional affinity matrix ($C \ll HW$) can achieve a self-attention with linear complexity. Similarly, linear attention [30] encodes a global representation on all features and selects relative semantic features from content sensing, which means that the channel querying on the compatibility matrix also makes sense for semantic representation. Efficient attention [31] has been proven effective on several kinds of visual tasks. Inspired by this, our flexible Transformer (FlexFormer) model for vision recognition tasks is built on the proposed Conv-MSA and linear attention mechanism, and has a similar pipeline with the pyramid vision Transformer model (PVT) [9]. It consists of four stages for feature encoding, and each stage contains two parts, an overlapped downsampling module (or stem block) and several Transformer encoder blocks, as seen in Fig. 1. In our Transformer encoder block, the proposed Conv-MSA is transformed into a channel queried self-attention block, whose computational complexity is linear to the input image size, followed by a locally enhanced Feed-forward network [32]. Finally, a full connection layer

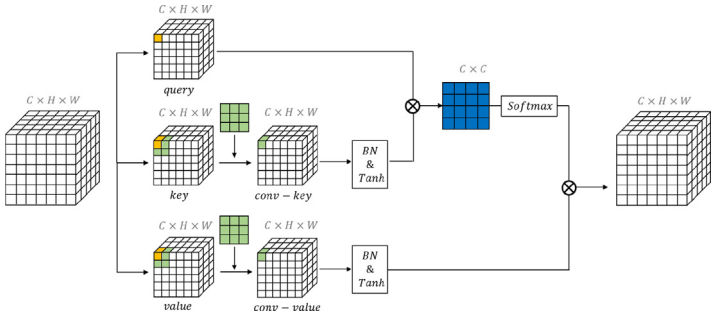


Fig. 4. Detailed architecture of proposed self-attention block.

maps the high-dimensional encoding features to a normalized confusion matrix of specific classes for our recognition tasks. In addition, Tanh-BatchNorm operations are performed on the depth-wise convolution applied *key* and *value* for fast convergence [33]. The self-attention module could be defined as,

$$\tilde{\mathbf{K}} = \text{Tanh}(\text{BatchNorm}(\text{DWConv}(\mathbf{K}))) \tag{6}$$

$$\tilde{\mathbf{V}} = \text{Tanh}(\text{BatchNorm}(\text{DWConv}(\mathbf{V}))) \tag{7}$$

$$\text{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q} \times \tilde{\mathbf{K}}^{\text{T}}}{\sqrt{HW}}\right) \times \tilde{\mathbf{V}} \tag{8}$$

where $\tilde{\mathbf{K}}$ and $\tilde{\mathbf{V}}$ are fine-grained information perceiving $\mathbf{K}$ and $\mathbf{V}$ and are defined in Eqs. (6) and (7), respectively. Compared with the previous self-attention modules, we will show that ours is not only effective, but also memory and calculation friendly. To enhance reproducibility, we provide the example code of our proposed self-attention module in Algorithm 1. Code is implemented with Pytorch.

Algorithm 1 PyTorch snippet of LCA.

```
import torch

Class LCA:
  def __init__(dim, head):
    self.qkv = torch.nn.Conv2d(dim, dim * 3, kernel_size=1)
    self.head = head
    self.dwconv_k = torch.nn.Conv2d(dim, dim, kernel_size=3, padding=1, groups=dim)
    self.dwconv_v = torch.nn.Conv2d(dim, dim, kernel_size=3, padding=1, groups=dim)
    self.bn_k = torch.nn.BatchNorm2d(dim)
    self.bn_v = torch.nn.BatchNorm2d(dim)
    self.act = torch.nn.Tanh()
    self.proj = torch.nn.Conv2d(dim, dim, kernel_size=1)
  def forward(x):
    B, C, H, W = x.shape
    scale = (H * W) ** -0.5
    qkv = self.qkv(x).reshape(B, 3, C, H, W).transpose(0, 1)
    q, k, v = qkv[0], qkv[1], qkv[2]
    q = q.reshape(B, self.head, C // self.head, -1)
    k = self.act(self.bn_k(self.dwconv_k(k))).reshape(B, self.head, C // self.head, -1)
    v = self.act(self.bn_v(self.dwconv_v(v))).reshape(B, self.head, C // self.head, H, W)
    attn = (q @ k.transpose(2, 3)) * scale
    attn = attn.softmax(dim = -1)
    out = attn @ v
    out = self.proj(out)
    return out
```

3. Experiments and discussion

We conduct abundant experiments to verify the effectiveness of our proposed Conv-MSA and FlexFormer. We present the main results on ImageNet [17] and compare them with various architectures. We also test our models on the downstream transfer learning datasets including CIFAR-10/100 [18], Oxford Pets [34], CK+ [35], JAFFE [36] and so on. Lastly, we investigate the efficiency and robustness of the proposed models and provide visualization to have an intuitive understanding of our method.

3.1. On ImageNet

Experimental settings The first experiment was conducted on the ImageNet [17] benchmark, which contains 1.28M training images and 50K validation images of 1000 classes. For a fair comparison, we adopt the data augmentation methods following DeiT [11] and do not use extra augmentation methods. The default batch size and initial learning rate are set to 1024 and $1e^{-3}$, and a cosine learning rate scheduler with 20 epochs linear warm-up is used. We train the model for 300 epochs using AdamW optimizer [37]. For 384×384 input We fine-tuned the models for 30 epochs with the weight decay of $1e^{-8}$, learning rate of $1e^{-5}$, and batch size of 512. The compared algorithms are all competitive models, including DeiT [11], CvT [22], PVT [9], Swin Transformer [6], ResT [7], TNT [38], Twins [16], CPVT [21], NomMer [15], Uniformer [39] and so on. All models are trained on 8 TITAN RTX GPUs. We designed five different architectures, FlexFormer-T, FlexFormer-S, FlexFormer-M, FlexFormer-L and FlexFormer-VL, following the scaling strategy. Each architecture begins with a 7×7 convolution operation and downsamples input to 56×56 scale, followed by four stages of transformer blocks consisting of Linear Conv-MSA and LFF module. After the first three stages the feature size is downsampled to 1/4 and channel number is doubled. Channel number for the first stage of five architectures is 48, 64, 64, 80 and 96 respectively. Similar to ResNet, we repeated the third stage most to extract abundant feature in five architectures, whose number is five times of other stages.

Results Table 1 summarizes the comparison of FlexFormer with other models. It can be seen that our FlexFormer outperforms previous convolution-enhanced SA models with a smaller model scale. Compared with CvT [22], our FlexFormer-S outperforms CvT-13 by 0.4% with nearly 15% parameter reduction and 40% FLOPs decreasing. Confronted with SOTA models, FlexFormer is still competitive in terms of accuracy-efficiency trade-off. For instance, FlexFormer-L outperforms UniFormer-B [39] by 0.5%, NomMer-S [15] by 0.7%, ResT-L [7] by 0.8% and Swin-B [6] by 0.9% with similar or smaller model size. In particular, FlexFormer-VL reaches the SOTA performance of 85.4% recognition accuracy with about 20% fewer param-

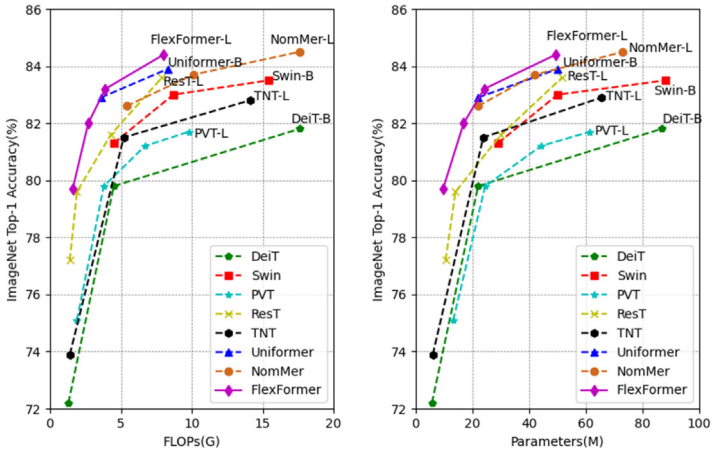


Fig. 5. Performance-Efficiency comparisons on 224 × 224**input size.** FlexFormer (our method) demonstrates better accuracy efficiency trade-off compared with state-of-the-art baselines. We report the ImageNet [17] Top-1 accuracy (*y*-axis) trade-off with respect to floating point operations (left) and parameter counts (right). Other metrics are from the original publications [6,7,9,11,15,38,39].

Table 1
Image classification performance on the ImageNet validation set. "Params." refers to the number of parameters, "In. S." indicates the size of input images, and "Ref." means the referenced work.

Method	Top-1 Acc. ↑	FLOPs ↓	Params. ↓	In. S.	Ref.
CPVT-Ti-GAP	74.9%	1.4G	6.0M	224^2	CVPR22
PVT-T	75.1%	1.9G	13.2M	224^2	ICCV21
ResT-S	79.6%	1.9G	13.6M	224^2	NIPS21
FlexFormer-T	**79.7%**	**1.6G**	**9.6M**	**224^2**	
PVT-S	79.8%	3.8G	24.5M	224^2	ICCV21
Swin-T	81.3%	4.5G	29.0M	224^2	ICCV21
TNT-S	81.5%	5.2G	23.8M	224^2	NIPS21
CvT-13	81.6%	4.5G	20.0M	224^2	ICCV21
Twins-SVT-S	81.7%	2.9G	24.0M	224^2	NIPS21
FlexFormer-S	**82.0%**	**2.7G**	**16.9M**	224^2	
DeiT-S	79.8%	4.5G	22.0M	224^2	ICML21
PVT-M	81.2%	6.7G	44.2M	224^2	ICCV21
CPVT-S-GAP	81.5%	4.9G	23.0M	224^2	CVPR22
ResT-B	81.6%	4.3G	30.1M	224^2	NIPS21
CvT-21	82.5%	7.1G	32.0M	224^2	ICCV21
NomMer-T	82.6%	5.4G	22.0M	224^2	CVPR22
UniFormer-S	82.9%	3.6G	22.0M	224^2	ICLR22
Swin-S	83.0%	8.7G	50.0M	224^2	ICCV21
Twins-SVT-B	83.2%	8.6G	56.0M	224^2	NIPS21
FlexFormer-M	**83.2%**	**3.9G**	**24.1M**	**224^2**	
PVT-L	81.7%	9.8G	61.4M	224^2	ICCV21
DeiT-B	81.8%	17.6G	86.7M	224^2	ICML21
CPVT-B	82.3%	18.3G	88.0M	224^2	CVPR22
TNT-B	82.8%	14.1G	65.5M	224^2	NIPS21
Swin-B	83.5%	15.4G	88.0M	224^2	ICCV21
ResT-L	83.6%	7.9G	51.6M	224^2	NIPS21
Twins-SVT-L	83.7%	15.1G	99.2M	224^2	NIPS21
NomMer-S	83.7%	10.1G	42.0M	224^2	CVPR22
UniFormer-B	83.9%	8.3G	50.0M	224^2	ICLR22
FlexFormer-L	**84.4%**	**8.0G**	**49.3M**	**224^2**	
DeiT-B	84.5%	55.4G	87.0M	384^2	ICML21
Swin-B	84.5%	47.0G	84.7M	384^2	ICCV21
NomMer-B	84.9%	56.2G	73.0M	384^2	CVPR22
FlexFormer-VL	**85.4%**	**34.1G**	**70.7M**	**384^2**	

eters than Swin-B [6]. Moreover, FlexFormer outperforms most recent mainstream models with fewer parameters and less computational cost. The performance-efficiency comparisons are shown in Fig. 5.

3.2. Transfer learning

Image Classification Transfer learning experiments from the pre-trained model on ImageNet to other downstream image classification tasks are further conducted to validate the generalization capability of FlexFormer. These datasets includes pervasive object recognition datasets (CIFAR10 [18], CIFAR100 [18]) and specific object recognition datasets (Stanford Cars [40], Oxford Flower [41], Oxford Pets [34]). The size of the training set is ranging from 2040 to 50,000. Overall datasets, each input image is resized to 384 × 384. The top-1 recognition accuracy results are listed in Table 2. It can be seen from the results that our model outperforms other vision Transformers with fewer FLOPs or parameters, which indicates our model has better-generalized representation capability.

Facial Expression Recognition Facial expression recognition (FER) has been regarded as a challenging task due to significant ambiguity in the expression of different subjects with different intensities. Experiments on FER tasks are compared with both previous CNN-based and Transformers methods. A pre-trained checkpoint on ImageNet is utilized to fine-tune our model. As illustrated in Table 3, experimental results show that with 10-fold cross-validation the FlexFormer-L achieves the SOTA performance of 94.0% accuracy on JAFFE [36], about 1.1% higher than ViT [42], and obtains 99.31% top-1 accuracy on CK+ [35], which overpasses the SOTA by 0.15% [43].

3.3. Ablation study

We design various ablation experiments to comprehensively explore the effectiveness of our proposed architecture and the trade-off between performance and efficiency. First, we show the effectiveness of our proposed Conv-MSA. Then, we investigate the impact of each component in FlexFormer. *Proposed Conv-MSA* - We validate the effectiveness of our proposed Conv-MSA based on DeiT [11]. Table 4 illustrates its priority. Compared with DeiT [11] and DeiT-based CPVT [21], our Conv-MSA not only outperforms them in small models (2.7% higher than DeiT-T and 1.5% higher than CPVT-T), but also shows superiority with model depth increasing (0.8% higher than DeiT-B and 0.3% higher than CPVT-B) with little cost. From visualization results of feature extraction evolution illustrated in Section 2.3, it could be concluded that Conv-MSA mitigates the shortage of MSA to some extent and shows competitive performance.

Component analysis - To demonstrate the contribution each component makes in the model, we conduct component analysis experiments based on PVT baseline [9], as shown in Table 5. Especially, our linear Conv-MSA can boost the recognition accuracy of PVT-T by 0.9% with much less parameter and computational cost.

Table 2
Transfer learning results on downstream classification tasks.

Model	FLOPs↓	Params↓	CIFAR10 ↑	CIFAR100 ↑	Cars ↑	Flowers ↑	Pets ↑
ResNet-152	11.3G	58.1M	97.9%	87.6%	92.0%	97.4%	94.5%
Inception-v4	16.1G	41.1M	97.9%	87.5%	93.3%	98.5%	93.7%
EfficientNet-B7	37.2G	64.0M	98.9%	91.7%	94.7%	98.8%	95.4%
ViT-B/16	17.6G	85.8M	98.1%	87.1%	-	89.5%	93.8%
DeiT-B	17.6G	85.8M	99.1%	90.8%	92.1%	98.4%	-
CeiT-S	12.9G	24.2M	99.1%	90.8%	94.1%	98.6%	94.9%
TNT-S	17.3G	23.8M	98.7%	90.1%	-	98.9%	94.7%
FlexFormer-M	**12.3G**	**24.1M**	**99.2%**	**91.9%**	**94.6%**	**99.0%**	**95.1%**

Table 3
Recognition accuracy on CK+ and JAFFE with 10-fold cross validation.

Dataset	Method	Top-1 Acc. ↑
CK+	PPDN [44]	97.30%
	DeepMotion [45]	98.00%
	FN2EN [46]	98.60%
	DDL [43]	99.16%
	FlexFormer-L	**99.31%**
JAFFE	Fisherface [47]	89.3%
	Salient Facial Patch [48]	91.8%
	DeepMotion [45]	92.8%
	ViT+SE [42]	92.9%
	FlexFormer-L	**94.0%**

Table 4
Comparison of ViT and ViT with Conv-MSA.

Method	FLOPs ↓	Params ↓	Top-1 Acc. ↑
DeiT-T [49]	1.3G	5.7M	72.2
CPVT-T [21]	1.3G	6.0M	73.4(+1.2)
DeiT-T+Conv-MSA	1.3G	5.8M	74.9(+2.7)
DeiT-S [49]	4.6G	22.0M	79.8
CPVT-S [21]	4.6G	23.0M	80.5(+0.7)
DeiT-S+Conv-MSA	4.6G	22.1M	81.0(+1.2)
DeiT-B [49]	17.6G	86.6M	81.8
CPVT-B [21]	17.6G	88.0M	82.3(+0.5)
DeiT-B+Conv-MSA	17.6G	86.8M	82.6(+0.8)

Table 5
Ablation study on model components.

Method	Top-1 Acc. ↑	Params ↓	FLOPs ↓
PVT-T [9]	75.1%	13.2M	1.97G
+Overlapped-embedding	75.5%	13.4M	1.98G
+Linear Conv-MSA	76.3%	11.9M	1.95G
w/o shortcut	76.0%	11.4M	1.93G
+LFF	76.6%	12.0M	1.95G
w/o shortcut	75.7%	13.2M	1.97G

Table 6
Ablation study on fast convergence.

Activation function	Top-1 Acc.	Epoches to reach 75.0% ↓	Epoches to reach 79.0% ↓
ReLU	79.24%	177	289
ELU	79.39%	174	284
GELU	79.67%	165	278
Tanh	79.74%	159	275

Table 8
Comparison of inference speed of different models.

Model	FLOPs ↓	Throughput (Image/s) ↑	Top-1 Acc. ↑
Swin-S	8.7G	436.9	83.0%
PVT-M	6.7G	528.1	81.2%
TNT-S	5.2G	428	81.5%
ResT-Large	7.9G	429	83.6%
Twins-SVT-B	8.6G	469	83.2%
FlexFormer-M	**3.9G**	**697.0**	**83.2%**

Normalization and activation functions - We also perform ablation experiments to study the effectiveness of normalization and activation function after DWConv. We found that the linear attention without normalization was easy to collapse and converged slowly during training. If the BatchNorm and activate functions (i.e. ReLU [50], ELU [51], Tanh [52] and GELU [53] etc.) are utilized, the FlexFormer with Tanh-Softmax hybrid nonlinearization will converge more rapidly when training. In other words, the Tanh-Softmax hybrid nonlinearization would be helpful to build a deeper Transformer model as well. Experimental results in Table 6 illustrate that the normalized dot-products based on the Tanh-BatchNorm non-linearity in our self-attention could not only boost the recognition accuracy to a higher level but also make the model converge rapidly. Specifically, it needs fewer epochs of gradient descending to obtain the same validation results.

Linear attention - Ablation studies on linear attention mechanisms are shown in Table 7. We compare the original attention and linear attention on FlexFormer-T and FlexFormer-S. For the original attention, we perform two large kernel convolutions like PVT [9] to reduce the spatial resolution. Though sharing similar performance, linear attention has lower computational complexity in both FlexFormer-T and FlexFormer-S. Moreover, linear attention is more memory friendly compared with the original attention, consuming about 25% less memory storage.

3.4. Inference speed analysis

Inference speed (throughput, images processed per second) testing of comparable models is conducted on a physical GPU processor. According to Touvron et al. [11], the throughput is measured with the 224×224 input size on an NVIDIA V100 GPU and PyTorch platform. From the results in Table 8, it can be seen that FlexFormer is approximately 1.5 times faster than most previous Transformer models during inference.

Table 7
Ablation study on attention modes.

Model	Method	Params ↓	FLOPs ↓	Memory usage ↓	Top-1 Acc. ↑
Flexformer-T	Original attention	10.0M	1.8G	10395M	79.69%
	Linear attention	9.6M	1.6G	7452M	79.74%
Flexformer-S	Original attention	17.8M	2.7G	16286M	82.15%
	Linear attention	16.9M	2.7G	12741M	82.16%

4. Conclusion

We propose a flexible Transformer recognition model, where the problems of spatial over-smoothing are alleviated with lightweight convolutions embedded to linear computational self-attention layers. We believe that more novel self-attention methods to improve the representation ability of Transformers will surpass traditional CNNs in more fields in the future.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, NIPS, 2017.
[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: transformers for image recognition at scale, ICLR, 2021.
[3] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, NIPS, 2012.
[4] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, CVPR, 2016.
[5] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, CVPR, 2016.
[6] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, B. Guo, Swin transformer: hierarchical vision transformer using shifted windows, ICCV, 2021.
[7] Q. Zhang, Y. Yang, ResT: an efficient transformer for visual recognition, NIPS, 2021.
[8] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J.M. Alvarez, P. Luo, SegFormer: simple and efficient design for semantic segmentation with transformers, NIPS, 2021.
[9] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, L. Shao, Pyramid vision transformer: a versatile backbone for dense prediction without convolutions, ICCV, 2021.
[10] Y. Jiang, S. Chang, Z. Wang, TransGAN: two pure transformers can make one strong GAN, and that can scale up, NIPS, 2021.
[11] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, H. Jégou, Training data-efficient image transformers & distillation through attention, ICML, 2021.
[12] N. Park, S. Kim, How do vision transformers work? ICLR, 2022.
[13] P. Wang, W. Zheng, T. Chen, Z. Wang, Anti-oversmoothing in deep vision transformers via the fourier domain analysis: from theory to practice, ICLR, 2022.
[14] Z. Qin, P. Zhang, F. Wu, X. Li, FcaNet: frequency channel attention networks, ICCV, 2021.
[15] H. Liu, X. Jiang, X. Li, Z. Bao, D. Jiang, B. Ren, NomMer: nominate synergistic context in vision transformer for visual recognition, CVPR, 2022.
[16] X. Chu, Z. Tian, Y. Wang, B. Zhang, H. Ren, X. Wei, H. Xia, C. Shen, Twins: revisiting the design of spatial attention in vision transformers, NIPS, 2021.
[17] O. Russakovsky, J. Denget, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, ImageNet: a large-scale hierarchical image database, CVPR, 2009.
[18] A. Krizhevsky, G. Hinton, Learning Multiple Layers of Features from Tiny Images, Technical Report, Citeseer, 2009.
[19] Z. Peng, W. Huang, S. Gu, L. Xie, Y. Wang, J. Jiao, Q. Ye, Conformer: local features coupling global representations for visual recognition, ICCV, 2021.
[20] Y. Chen, X. Dai, D. Chen, M. Liu, X. Dong, L. Yuan, Z. Liu, Mobile-former: bridging mobilenet and transformer, CVPR, 2022.
[21] X. Chu, Z. Tian, B. Zhang, X. Wang, X. Wei, H. Xia, C. Shen, Conditional positional encodings for vision transformers, CVPR, 2022.
[22] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, L. Zhang, CvT: introducing convolutions to vision transformers, ICCV, 2021.
[23] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, J. Shlens, Stand alone self-attention in vision models, NIPS, 2019.
[24] I. Bello, LambdaNetworks: modeling long-range interactions without attention, ICML, 2021.
[25] Y. Cao, J. Xu, S. Lin, F. Wei, H. Hu, GCNet: non-local networks meet squeeze-excitation networks and beyond, CVPR, 2019.
[26] H. Liu, F. Liu, X. Fan, D. Huang, Polarized self-attention: towards high-quality pixel-wise regression, 2021. arXiv:2107.00782.
[27] M.A. Islam, S. Jia, N.D.B. Bruce, How much position information do convolutional neural networks encode? ICLR, 2020.
[28] A. Hassani, S. Walton, J. Li, S. Li, H. Shi, Neighborhood attention transformer, 2022. arXiv preprint arXiv:2204.07143.
[29] A.G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, MobileNets: efficient convolutional neural networks for mobile vision applications, 2017, arXiv preprint arXiv:1704.04861.
[30] R. Li, J. Su, C. Duan, S. Zheng, Linear attention mechanism: an efficient attention for semantic segmentation, 2020. arXiv:2007.14902.
[31] Z. Shen, M. Zhang, H. Zhao, S. Yi, H. Li, Efficient attention: attention with linear complexities, WACV, 2021.
[32] K. Yuan, S. Guo, Z. Liu, A. Zhou, F. Yu, W. Wu, Incorporating convolution designs into visual transformers, ICCV, 2021.
[33] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, ICML, 2015.
[34] O.M. Parkhi, A. Vedaldi, A. Zisserman, C.V. Jawahar, Cats and dogs, CVPR, 2012.
[35] P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, I. Matthews, The extended Cohn-Kanade Dataset (CK+): a complete dataset for action unit and emotion-specified expression, CVPR, 2010.
[36] M. Lyons, S. Akamatsu, M. Kamachi, J. Gyoba, Coding facial expressions with Gabor wavelets, in: IEEE International Conference on Automatic Face and Gesture Recognition, 1998.
[37] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, ICLR, 2019.
[38] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, Y. Wang, Transformer in transformer, NIPS, 2021.
[39] K. Li, Y. Wang, P. Gao, G. Song, Y. Liu, H. Li, Y. Qiao, UniFormer: unified transformer for efficient spatiotemporal representation learning, ICLR, 2022.
[40] J. Krause, M. Stark, J. Deng, L. Fei-Fei1, 3D object representations for fine–grained categorization, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2008.
[41] M.-E. Nilsback, A. Zisserman, Automated flower classification over a large number of classes, in: IEEE Conference on Computer Vision, Graphics & Image Processing, 2008.
[42] M. Aouayeb, W. Hamidouche, C. Soladie, K. Kpalma, Learning vision transformer with squeeze and excitation for facial expression recognition, 2021. arXiv preprint arXiv:2107.03107.
[43] D. Ruan, Y. Yan, S. Chen, J.-H. Xue, H. Wang, Deep disturbance-disentangled learning for facial expression recognition, in: Proceedings of the 28th ACM International Conference on Multimedia, 2020.
[44] X. Zhao, X. Liang, L. Liu, T. Li, Y. Han, N. Vasconcelos, Y. Shuicheng, Peak-piloted deep network for facial expression recognition, ECCV, 2016.
[45] S. Minaee, M. Minaei, A. Abdolrashidi, Deep-emotion: facial expression recognition using attentional convolutional network, Sensors 21 (2021) 3046.
[46] H. Ding, S.K. Zhou, R. Chellappa, FaceNet2ExpNet: regularizing a deep face recognition net for expression recognition, in: Proceeding of IEEE International Conference on Automatic Face & Gesture Recognition, 2017.
[47] Z. Abidin, A. Harjoko, A neural network based facial expression recognition using fisherface, Int. J. Comput. Appl. 56 (3) (2012) 30–34.
[48] S.L. Happy, A. Routray, Automatic facial expression recognition using features of salient facial patches, IEEE Trans. Affect. Comput. 6 (2015) 1–12.
[49] A. Srinivas, T.-Y. Lin, N. Parmar, J. Shlens, P. Abbeel, A. Vaswani, Bottleneck transformers for visual recognition, CVPR, 2021.
[50] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the 14th International Conference on Artificial Intelligence and Statistics, 2011.
[51] D.-A. Clevert, T. Unterthiner, S. Hochreiter, Fast and accurate deep network learning by exponential linear units (ELUs), ICLR, 2016.
[52] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in: Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, 2010.
[53] D. Hendrycks, K. Gimpel, Bridging nonlinearities and stochastic regularizers with gaussian error linear units, 2016. arXiv preprint arXiv:1606.08415.