ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240476

FMViT: A Multiple-Frequency Mixing Vision Transformer

Wei Tan[a], Yifeng Geng[a] and Xuansong Xie[a]

[a]Institute for Intelligent Computing, Alibaba Group
{ziyuan.tw, cangyu.gyf}@alibaba-inc.com, xingtong.xxs@taobao.com

Abstract. The transformer model has gained widespread adoption in computer vision tasks in recent times. However, due to the quadratic time and memory complexity of self-attention, which is proportional to the number of input tokens, most existing Vision Transformers (ViTs) encounter challenges in achieving efficient performance in practical industrial deployment scenarios, such as TensorRT and CoreML, where traditional CNNs excel. Although some recent attempts have been made to design CNN-Transformer hybrid architectures to tackle this problem, their overall performance has not met expectations. To tackle these challenges, we propose an efficient hybrid ViT architecture named FMViT. This approach enhances the model's expressive power by blending high-frequency features and low-frequency features with varying frequencies, enabling it to capture both local and global information effectively. Additionally, we introduce deploy-friendly mechanisms such as Multiple groups of MLP (gMLP) Reparameterization, Lightweight Multi-head Self-Attention (RLMHSA), and Convolutional Fusion Block (CFB) to further improve the model's performance and reduce computational overhead. Our experiments demonstrate that FMViT surpasses existing CNNs, ViTs, and CNN-Transformer hybrid architectures in terms of latency/accuracy trade-offs for various vision tasks. On the TensorRT platform, FMViT outperforms Resnet101 by 2.5% (83.3% vs. 80.8%) in top-1 accuracy on the ImageNet dataset while maintaining similar inference latency. Moreover, FMViT achieves comparable performance with EfficientNet-B5 but with a 43% improvement in inference speed. On CoreML, FMViT outperforms MobileOne by 2.6% in top-1 accuracy on the ImageNet dataset, with inference latency comparable to MobileOne (78.5% vs. 75.9%).

1 Introduction

Vision Transformers (ViTs) have recently succeeded in various computer vision applications such as image classification, object detection, and semantic segmentation and have received extensive attention from industry and academia. Despite this, Convolutional Neural Networks (CNNs) remain the preferred choice for real-world vision tasks, primarily because ViTs typically exhibit slower performance than traditional CNNs, such as ResNets. The inference speed of Transformer models is constrained by elements such as the Multi-head Self-attention (MHSA) mechanism, non-fusible LayerNorm, and GELU layers, along with frequent memory accesses.

Numerous endeavors have been undertaken to liberate Vits from the high-latency issue. For instance, models such as Swin [22], PoolFormer [37], Reformer [16], MaxViT [34], SepViT [18], and MobileViT [26], among others, strive to develop spatial attention methods that are more efficient and mitigate the quadratic surge in computational complexity of MHSA. Concurrently, other initiatives, including EfficientFormer [20] and MobileViT, are exploring ways to construct CNN-Transformer hybrid architectures that balance accuracy and latency. This is achieved by integrating effective convolutional blocks with potent Transformer blocks. Notably, most of the current state-of-the-art (SOTA) models are designed as CNN-Transformer hybrids. These models predominantly utilize convolutional blocks in the preliminary stages and reserve the stacking of Transformer blocks for the final stages.

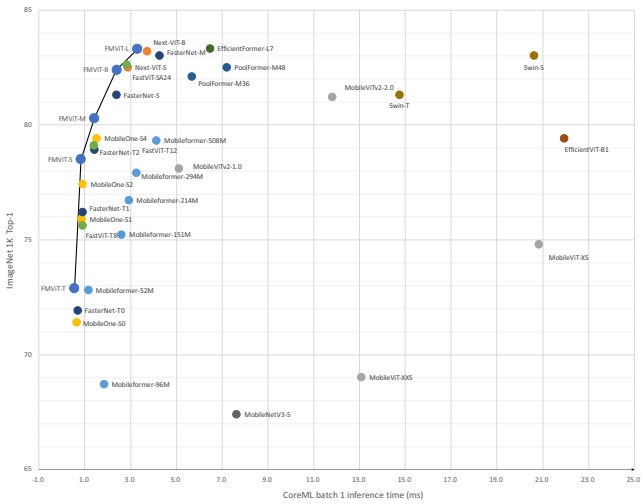

Figure 1. Speed-performance(classification accuracy with Top-1) trade-off on ImageNet1K

Presently, neither the Convolutional Block nor the Transformer Block can simultaneously achieve efficiency and performance in existing works. Although the precision-latency tradeoff has improved over the years, the overall performance of modern hybrid systems still needs to improve. This study introduces four critical components for designing effective vision Transformer networks to address these challenges. Firstly, inspired by NextViT's [17] mixing of high-frequency features and low-frequency features, a potent Multi-Frequency Fusion Block (FMB) is introduced, amalgamating multiple high-frequency and low-frequency features to enhance the model's information flow and expressive capacity. Secondly, a Lightweight Convolution Fusion Block (CFB) is proposed to efficiently blend the local modeling capability of convolution with convolution multi-group reparameterization, further bolstering modeling performance. Thirdly, convolutional multi-group reparameterization is suggested. It integrates the spatial information of different

subchannels during the training phase. It fuses them into a convolution in the inference phase, improving the model's accuracy while maintaining the inference speed. Lastly, a lightweight self-attention block, termed RLMHSA, is developed. It employs a lightweight and reparameterized design to augment the modeling ability and expedite the inference stage.

A CNN-Transformer hybrid architecture, FMViT, is introduced based on the above methods. Analogous to NextViT [17], the use of TensorRT and CoreML signifies real deployed architectures in server-side and mobile devices, respectively, with their inference latency representing the actual time consumption in the industry. As depicted in Figure 1, FMViT achieves an optimal balance between delay and accuracy in the ImageNet-1K classification task. On TensorRT, FMViT surpasses Resnet101 by 2.5% in top-1 accuracy on the ImageNet dataset, maintaining a comparable inference latency. Concurrently, it exhibits performance on par with EfficientNet-B5, enhancing the inference speed by 43%. On CoreML, the top-1 accuracy on the ImageNet dataset exceeds MobileOne by 2.6% while maintaining a similar inference latency.

Our major contributions are outlined below:

- An efficient Multi-Frequency Fusion Block (FMB) is proposed to combine multiple sets of high-frequency and low-frequency features, enhance the information flow of the model, and enhance the expression ability of the model.
- Proposes a lightweight Convolutional Fusion Block (CFB), which efficiently blends the local modeling capabilities of Convolutions and uses convolutional multi-group reparameterization to further provide modeling performance.
- Convolutional multi-group reparameterization is proposed, which fuses the spatial information of different subchannels in the training stage and fuses it into a convolution in the inference stage, so as to improve the accuracy of the model while the inference speed is unchanged.
- Multiple groups of Multilayer Perceptron Layer (gMLP) blocks were proposed to fuse global signals and local information to enhance the expression ability of the model.
- Proposes a Lightweight Self-Attention Block (RLMHSA), which adopts a lightweight and reparameterized design, enhances the global modeling ability of the module, and improves the speed of the inference stage.

2 Related Work

2.1 Convolutional Networks

Convolutional Neural Networks (CNNs) have been the de facto vision architecture standard for various computer vision applications, such as semantic segmentation, object identification, and image classification, since 2012. ResNet [8] uses residual connections to stop the network from deteriorating and keep the network deep and able to capture high-level abstractions. On the other hand, DenseNet [12] promotes the concatenation of feature maps and the reuse of features. Convolution is introduced point-wise and depth-wise by MobileNets [29] to create models with less memory usage and quicker response times. By using group point-wise convolution and channel shuffling, ShuffleNet [40] substantially lowers computing expenses. According to ShuffleNetv2 [25], network architecture design should prioritize direct metrics, such as speed, over indirect metrics, like FLOPs. ConvNeXt [23] explores the structure of vision Transformers and proposes a pure CNN model that, while retaining the simplicity and efficiency of traditional CNNs, can effectively compete with state-of-the-art hierarchical vision Transformers across a range of computer vision benchmarks.

2.2 Vision Transformers

The concept of the Transformer was initially introduced in the field of natural language processing (NLP). ViT [7], in implementing self-attention, segments the image into patches and treats these patches as words, thereby demonstrating the Transformer's efficacy in various vision-related tasks. The teacher-student method proposed by DeiT [33] is designed explicitly for Transformers. T2T-ViT [38] introduces a unique token-to-token process to progressively tokenize images into tokens and aggregate them structurally. The Swin Transformer introduces a universal backbone that constructs hierarchical features with a computational cost linearly proportional to the image size. Meanwhile, PiT [11] incorporates a pooling layer into ViT and conducts experiments to validate its efficacy.

2.3 Hybrid Models

Recent studies indicate that a hybrid design [20, 30, 26], integrating both convolution and Transformer, effectively leverages the strengths of both architectures. BoTNet [30] employs global self-attention to supplant the spatial convolutions of the final three bottleneck blocks in ResNet. Concurrently, lightweight and efficient ViTs, such as MobileViT [26] and MobileViTv2 [27], have been introduced for mobile devices. The fusion of Mobile-Former [2] with the proposed lightweight cross-attention model enhances computational efficiency and boosts representational power. EfficientFormer [20] and EfficientFormerV2 [19] adhere to size-consistent designs, seamlessly employing hardware-friendly 4D MetaBlocks and potent 3D MHSA blocks for joint size and speed search via NAS. ToMe [1] presents a ViT model that accelerates without training. BiFormer establishes an efficient pyramid network architecture through bidirectional attention routing. NextViT [17] captures one high-frequency feature and one low-frequency feature in the network separately, which are then blended to enhance the modeling capabilities of the model.

2.4 Structural Reparameterization

Reparameterization employs complex modules to enhance model performance during the training phase. It consolidates these complex modules into simpler ones during the inference phase, following the linear principle of the convolution operator. This process aims to boost the model's inference speed without compromising performance. ACNet [3] pioneered reparameterization to merge 3x3 convolutions into a 1x1 convolution, while RepVGG [6] applied reparameterization to skip connections, thereby reducing memory access costs. DBB [4] further expanded upon six prevalent reparameterization methods. The concept of linear training time overparameterization was introduced to augment the power of such models [13]. MobileOne [35] employs over-parameterization to enhance the performance of vision converters for mobile devices.

3 Methods

This section introduces the proposed FMViT architecture, followed by a discussion and analysis of its key designs. These include

the Convolutional Fusion Block (CFB), the Multi-Frequency Fusion Block (FMB), the Lightweight Multi-head Attention Module (RLMHSA), the Convolutional Multi-group Reparameterization method (gMLP), and the MLP module constructed using this method.

3.1 Overview

Figure 2 illustrates the comprehensive FMViT architecture. FMViT utilizes a conventional pyramidal structure, with each stage comprising a downsampling module and a Convolutional Fusion Block (CFB) with convolution only or a Multi-Frequency Fusion Block (FMB) with transformers. The stem reduces the spatial resolution to a quarter of the original input image, and each subsequent stage progressively halves the spatial resolution while incrementally increasing the number of channels. We explore the information interaction module and, inspired by MetaFormer [37], introduce the Convolutional Fusion Block (CFB) to address short-range data dependencies. A Multi-Frequency Fusion Block (FMB) is proposed to further fuse local and global data by decomposing multiple frequency bands. This multi-channel frequency fusion enhances the model's capacity for modeling. To decrease the computational complexity of Multi-head Attention (MHSA), we propose a lightweight RLMHSA module. The model's inference speed is improved without significantly compromising accuracy through parameter sharing and re-parameterization. These core modules develop a series of CNN-Transformer hybrid architectures that achieve an optimal balance between accuracy and latency on mobile-side CPUs and server-side GPUs, surpassing the state-of-the-art model.

3.2 Convolutional multi-group reparameterization

The 1x1 convolution is a linear fusion or channel conversion mechanism with global modeling capacity. The translation invariance characteristic of the kxk convolution is utilized to depict local space. The lack of local modeling between specific neighboring channel features limits ungrouped convolution operators' efficient information fusion capabilities. During the training phase, we suggest group reparameterization of kxk convolutions, initially defining the convolution operation as CONV(K_h,K_w,G), where K_h and K_w represent the convolution kernel sizes, and G denotes the convolution group size.

Assuming that the original convolution is defined as CONVA = CONV(K_h, K_w, G), during the training phase, multiple convolutions with diverse group sizes are connected in parallel:

$$CONVB = CONVA + \sum_{i=1}^{N} CONV(K_{hi}, K_{wi}, G_i) \quad (1)$$

Where $\forall i \in N, G_i >= G, K_{hi} <= K_h, K_{wi} <= K_w$, and N is a predefined constant.

Figure 3 illustrates the reparameterization of CONVB into CONVA during the inference phase. Any CONV(K_{hi}, K_{wi}, G_i) convolution is equivalent to the sparse CONV(K_h, K_w, G) convolution, signifying that the weight of the dotted line in the figure remains at a constant zero, while the other weights are unchanged. Based on additivity [4], the inference stage for two convolutions with the same number of groups can be reparameterized as the convolution CONVA, as depicted in the lower portion of Figure 3. Here, the left side represents the training stage, and the right side represents the inference stage, both of which are equivalent. Convolutional multi-group reparameterization enhances model performance without affecting the primary model's inference time.

RepMLP [5] proposes the reparameterization of the multilayer perceptron (MLP). RepMLP employs convKxK for fusion into FC, but its weight conversion is sparse, and the parameters post-conversion become KxK times the original, rendering it unsuitable for lightweight scenarios. In the Transformer, MLPs significantly contribute to performance due to their global modeling capabilities. However, the absence of local modeling capability restricts its potential. To aid the original convolutions in grouping modeling of local channels, multiple parallel conv1x1 convolutions with $G' > 1$ are introduced to the two original CONV(1,1,1) convolutions, reconstructing the MLP module. This approach concurrently focuses on information from different representation subspaces at various locations for efficient local representation learning.

$$CONV^{'}X(1,1,1) = CONVX(1,1,1) + \sum_{i=1}^{N} CONV(1,1,G_i) \quad (2)$$

In this context, X=1 or X=2 signifies either CONV1 or CONV2. CONV1 and CONV2 represent the two pre-reparameterized convolutions of the MLP, while the post-reparameterized convolutions of the MLP are also denoted as $CONV^{'}1$ and $CONV^{'}2$.

To enhance the hybrid modeling of global and local aspects in MLP, a depthwise convolution is incorporated between the two conv1x1s. A shortcut connection ensures that the global and local information flows do not interfere, and the added depthwise convolution is reparameterized. Experimental results indicate that augmenting the depthwise convolution capacity for local information fusion enhances MLP performance by 1.96% on Imagenet1k.

3.3 Convolutional Fusion Block(CFB)

Transformer blocks have demonstrated significant results across various vision tasks, with the attention-based token mixer module and the MetaFormer [37] paradigm underscoring their inherent advantages. However, the inference speed of Transformer blocks could be more efficient, particularly on mobile devices where the performance of multi-head attention, LayerNorm, and GELU calculations is suboptimal. We have demonstrated the success of the MetaFormer paradigm and, building upon it, propose an efficient CFB module that exclusively employs depthwise separable convolutions (DWConv) as the token mixer. CFB maintains the deployment advantage of the BottleNeck block while achieving comparable performance to a Transformer block, as depicted in Figure 2. The CFB is constructed using DWConv and MLP, adhering to the general MetaFormer design. CFB ensures performance while significantly improving inference deployment performance. Additionally, reparameterization is implemented during training to enhance further CFB's performance, where DWConv utilizes a standard, widely-used reparameterization. The MLP employs the convolutional multigroup reparameterization proposed in this work.

3.4 Multi-frequency Fusion Block(FMB)

While CFB has effectively learned local representation, the pressing need to handle global information collection remains. Transformer blocks capture low-frequency signals, providing global information such as shape and structure. Existing research indicates that Transformer blocks may partially diminish high-frequency information, including local texture details. To extract more fundamental and unique features, signals from various frequency bands must be meticulously integrated, as they are crucial to the human visual system.

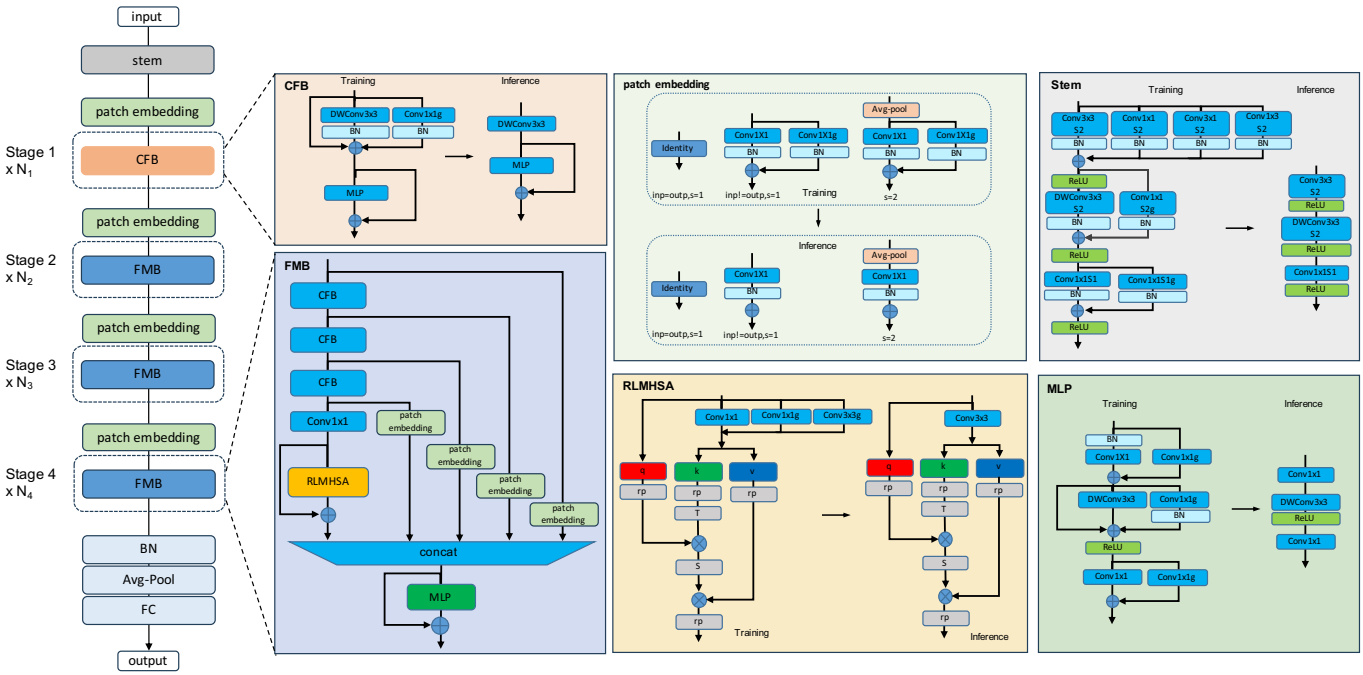

Figure 2. The figure shows the overall structure of FMViT. It mainly includes the Convolutional Fusion Block (CFB), the Multi-Frequency Fusion Block (FMB), the lightweight Multi-head Attention Module (RLMHSA), and the parameterized Multi-layer Perceptron Module (gMLP).

High-frequency signals provide local information, which is integral to preserving the completeness of the information. Distinct frequency features contribute unique information, rendering high-frequency signals vulnerable to degradation by the Transformer Block. The amalgamation of various high-frequency characteristics with low-frequency features could enhance the model's information flow and expressive capacity, drawing inspiration from information distillation and frequency mixing in image super-resolution [14]. As illustrated in Figure 2, the CFB module initially captures high-frequency features, subsequently outputting three sets of high-frequency features at different frequencies. Patch embedding fusion is then employed to splice the output of the lightweight multi-head attention module, creating a signal with both high and low frequencies. Through MLP layers, more fundamental and pronounced traits are extracted. The following formula can be expressed as follows:

$$
\begin{aligned}
z_1 &= f_1(x^{l-1}) \\
z_2 &= f_2(z_1) \\
z_3 &= f_3(z_2) \\
z_4 &= f_4(z_3) \\
z &= CONCAT(x^{l-1}, z_1, z_2, z_3, z_4) \\
x^l &= z + MLP(z)
\end{aligned}
$$

Here, x^{l-1} is defined as the input of the $(l-1)$th block, while x^l signifies the output of the lth block. CONCAT refers to the CAT join operation. f_1–f_3 represent high-pass filters that generate different high-frequency signals, as exemplified by CFB. f_4 is the low-pass filter that produces the low-frequency signal, as demonstrated by RLMHSA.

Unlike LN and GELU, FMB consistently employs BN and ReLU as the effective norm and activation layers [17]. These operators can be efficiently computed, especially on mobile devices, with minimal performance loss due to their speed-friendly nature.

FMB can gather and integrate multi-frequency information within a lightweight framework, thereby significantly enhancing the model's performance compared to the traditional pure Transformer module.

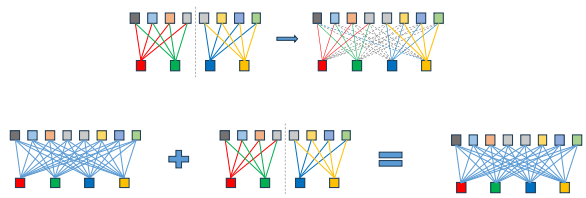

Figure 3. Top: A schematic depiction illustrating the notion of convolutional multi-group reparameterization, CONV(K_{hi}, K_{wi}, G_i) is comparable to sparse CONV(K_h, K_w, G). Bottom: By reparameterizing numerous groups of convolutions in the training phase, different groups of convolutions in the training phase are equal to a single convolution in the inference phase.

3.5 *Lightweight Multi-head Self-Attention (RLMHSA)*

The computational demand of the Transformer is proportional to the square of the input token dimension, making it computationally intensive when dealing with large input dimensions. Despite its relatively small number of parameters, the inference time on mobile devices, for instance, is extensive, necessitating a more lightweight design for the self-attention module. ViT-LSLA [10] substitutes the Key (K) and Value (V) of self-attention with the original input (X) to achieve a lightweight self-attention structure. As depicted in Figure 2, we propose a lightweight multi-head self-attention method that shares parameters and then applies reparameterization in this study.

The original MSA is defined as follows:

$$Atten(Q, K, V) = Softmax(QK^T)V \tag{3}$$

Where $Q = XW_q$, $K = XW_k$, and $V = XW_v$, input $X \in R^{M\times d}$, Query, Key, and Value matrices W_q, W_k, and $W_v \in R^{d\times d}$, respectively, M is the number of tokens, and d is the dimension of tokens. By deforming Equation 3, we obtain:

$$\begin{aligned} Atten(Q, K, V) &= Softmax(XW_q(XW_k)^T)XW_v \\ &= Softmax(XW_qW_k^TX^T)XW_v \\ &= Softmax(XW_{qk}X^T)XW_v \\ &= Softmax(X(XW_{qk}^T)^T)XW_v \\ &= Atten(X, K', V) \end{aligned}$$

The projection matrices of Q and K are consolidated for parameter sharing, which equates to a new matrix W_{qk} with $K = XW_{qk}^T$. Moreover, allowing W_{qk} and W_v to share parameters, they share a projection convolution, denoted as $W = W_{qk}^T = W_v$, then:

$$Atten(X, K', V') = Softmax(X(XW)^T)XW \tag{4}$$

Where $K' = XW, V' = XW$, and the structure is shown in Figure 2 for RLMHSA. Consequently, a single convolution is required to map the input vector, with the K and Q vectors sharing the same convolution, thereby eliminating the need for two separate convolutions. During training, convolutional multi-group parameterization is employed to emulate the blend of local and global information that characterizes the RLMHSA module and enhances MHSA performance. Experimental results indicate that, compared to MHSA on the Imagenet1k classification task, RLMHSA reduces the parameter count by 8M, accelerates speed by 3%, and enhances the top-1 classification accuracy by 0.5%.

3.6 Stem block and Patch Embedding block

The model's initial stage employs a stem with two down-sampling operations to reduce computational load, as suggested by FastViT [36]. A lightweight structure is achieved through the use of a Conv3x3+DWConv3x3 design. Initial convolutions utilize Conv3x1 and Conv1x3 to reparameterize vertical and horizontal edge extraction.

Several scenarios exist for patch embedding: no operation is necessary if the input and output channel numbers and token dimensions are identical. Conv1x1 is employed for channel number conversion when the input and output channel numbers differ, but the token dimension remains the same. If the token dimensions vary, a lightweight downsampling operation, avg-pool, is utilized for downsampling, followed by a Conv1x1 convolution for fusion or transformation. During the training phase, convolutional multi-group reparameterization is also applied to enhance accuracy.

3.7 FMViT Architectures

As delineated in Table 1, this study presents five model structures for reference based on the number of parameters and model size, specifically FMViT-T, FMViT-S, FMViT-M, FMViT-B, and FMViT-L. Here, "Channels" refers to the number of output channels from the internal submodule of the module; "FM-Channels" denotes the number of FMB intermediate frequency division channels, and "S" represents the stride in convolution, or Avg-pool. The expansion ratio for each MLP layer is set at 2, and the head dimension in RLMHSA is fixed at 32. In alignment with Nextvit [17], BatchNorm, and ReLU are employed for normalization and activation functions.

Table 1. Architecture details of FMViT variants.

Stages	Output size	Layers		FMViT-T	FMViT-S	FMViT-M	FMViT-B	FMViT-L
Stem	(H/4, W/4)	CNN Layers		Conv3x3, S=2				
				DWConv3x3, S=2				
				Conv1x1, S=1				
				(32,32,32)	(48,48,48)	(64,64,64)		
Stage1	(H/4, W/4)	Patch Embedding		Conv1x1, S=1				
				32	48	64		
		CFB	Channels	(32,32,32)	(48,48,48)	(64,96,96)		
			Blocks	3	3	3	3	6
Stage2	(H/8, W/8)	Patch Embedding		Avg-Pool,S=2				
				Conv1x1,S=1				
				32	48	96		
		FMB	Channels	(32,64,80)	(48,96,160)	(96,128,160)	(96,256,320)	
			FM-Channels	16	32	32	64	
			Blocks	1	1	1	1	
Stage3	(H/16, W/16)	Patch Embedding		Avg-Pool,S=2				
				Conv1x1,S=1				
				80	160	160	320	
		FMB	Channels	(80,128,160)	(160,192,320)	(160,320,480)	(320,384,480)	
			FM-Channels	32	64	96	96	
			Blocks	1	1	1	2	5
Stage4	(H/32, W/32)	Patch Embedding		Avg-Pool,S=2				
				Conv1x1,S=1				
				160	320	480		
		FMB	Channels	(160,192,320)	(320,384,640)	(480,512,960)	(480,640,1280)	
			FM-Channels	64	128	192	256	
			Blocks	1	1	1	1	

4 Experimental Results

In this experiment segment, we utilize PyTorch version 1.12.1 for PyTorch inference latency and the TensorRT-8.5.3 framework for TensorRT (TRT) inference latency. Both are measured in a hardware environment with an A10 GPU and CUDA 11.3. CoreML inference latency is gauged using an iPhone 13 with iOS 16.6. All batch sizes are uniformly set to 1.

4.1 ImageNet-1K Classification

4.1.1 Implementation

We executed an image classification experiment on the ImageNet-1K [28] dataset, comprising approximately 1.28 million training images and 50,000 validation images across 1,000 categories. To maintain fairness, we replicated the training parameters of the most recent vision Transformer with minor adjustments. All FMViT variants underwent training on eight V100 GPUs with a total batch size 2048 for 300 iterations. The input image was resized to a resolution of 224 x 224. Utilizing a weight decay of 0.1, we employed the AdamW [24] optimizer. For all FMViT variants, the learning rate was gradually reduced based on the cosine strategy, starting at 4e-5, and a linear warm-up approach was used for 20 epochs.

4.1.2 Comparison with State-of-the-art Models

As illustrated in Table 2, our method achieves an optimal balance of accuracy and latency when juxtaposed with recent state-of-the-art techniques such as CNNs, ViTs, and hybrid networks. When benchmarked against renowned CNNs like ResNet101 [8], FMViT surpasses ResNet101 by 2.5% in top-1 accuracy on the ImageNet dataset (83.3% vs. 80.8%) and is 45% faster on CoreML (3.5 ms vs. 2.4 ms). Concurrently, its performance is on par with EfficientNet-B5 [31], with a 43% improvement in inference speed. In the context of ViT, FMViT-B outperforms Swin-T [22] by being up to 6x

faster on CoreML and TensorRT, yet with a 1.1% superior performance. FMViT-S exceeds DeiT-T [32] by 6.3% on TensorRT at an equivalent inference speed (78.5% vs. 72.2%). FMViT-S surpasses CoreML by 8.1 percent (80.3% vs. 72.2%) with a similar inference speed. FMViT-L matches the performance of EfficientFormer-L7 [20] when compared to the hybrid approach, but the inference is 30% and 96% faster on TensorRT and CoreML, respectively. FMViT-S achieves 2.6% higher performance (78.5% vs. 75.9%) with a comparable CoreML inference speed when compared to MobileOne-S1 [35], and CoreML achieves 11% faster inference performance while achieving 2.9% higher accuracy (78.5% vs. 75.6%). These results suggest that the proposed FMViT design is a feasible and promising paradigm.

Table 2. Comparison of different state-of-the-art classification methods for ImageNet-1K.

Method	Image Size	Param (M)	FLOPs (G)	Latency(ms) pytorch	TRT	CoreML	Top-1 (%)
MobileViT-XXS	224	1.3	0.3	8.1	1.2	13.10	69.0
Mobileformer-52M	224	3.5	0.1	13.3	2.9	1.17	72.8
Mobileformer-96M	224	4.6	0.1	14.3	3.1	1.86	68.7
RepVGG-A0	224	8.3	1.4	2.1	0.7	1.06	72.4
MobileNetV1	224	4.2	0.6	2.2	0.5	0.84	70.6
MobileNetV3-S	224	2.5	0.1	5.0	0.7	7.65	67.4
FasterNet-T0	224	3.9	0.3	5.1	0.8	0.72	71.9
MobileOne-S0	224	2.1	0.3	2.5	0.4	0.65	71.4
MobileNetV2×1.0	224	3.5	0.3	5.0	0.7	0.87	71.8
DeiT-T	224	5.9	1.2	5.1	1.2	1.68	72.2
FMViT-T	**224**	**2.0**	**0.3**	**6.7**	**0.8**	**0.55**	**72.9**
RepVGG-B1	224	51.8	11.8	3.1	2.5	4.12	78.4
RepVGG-A2	224	25.5	5.1	2.4	1.4	2.11	76.5
Mobileformer-151M	224	7.6	0.2	18.9	4.3	2.61	75.2
Mobileformer-214M	224	9.4	0.2	20.1	4.6	2.93	76.7
Mobileformer-294M	224	11.4	0.3	19.5	5.0	3.27	77.9
FasterNet-T1	224	7.6	0.9	5.4	0.9	0.93	76.2
MobileOne-S1	224	4.8	0.8	2.4	0.6	0.87	75.9
MobileOne-S2	224	7.8	1.3	2.5	0.6	0.92	77.4
EfficientNet-B0	224	5.3	0.4	7.9	1.1	1.64	77.1
FastViT-T8	256	3.6	0.7	4.9	1.3	0.92	75.6
EfficientFormerV2-S0	224	3.5	0.4	10.2	1.3	0.89	75.7
MobileViTv2-1.0	256	4.9	1.8	8.3	2.4	5.12	78.1
MobileViT-XS	224	2.3	0.8	8.6	1.4	20.84	74.8
FMViT-S	**224**	**6.4**	**0.8**	**7.0**	**1.1**	**0.83**	**78.5**
FasterNet-T2	224	15.0	1.9	5.1	1.2	1.44	78.9
Mobileformer-508M	224	14.0	0.5	19.9	5.7	4.14	79.3
EfficientNet-B1	224	7.8	0.6	11.4	1.6	2.08	79.1
EfficientViT-B1	224	9.1	0.5	8.8	0.7	21.95	79.4
FastViT-T12	256	6.8	1.4	6.0	1.7	1.42	79.1
MobileOne-S4	224	14.8	3.0	4.5	1.2	1.52	79.4
DeiT-S	224	22.0	4.6	5.2	2.0	3.74	79.8
PoolFormer-S24	224	21.1	3.4	9.8	3.9	2.45	80.3
ResNeXt101-32x4d	224	44.2	8.0	13.5	3.9	3.65	78.8
EfficientFormer-L1	224	12.3	1.3	6.9	1.3	1.57	79.2
FMViT-M	**224**	**12.8**	**2.0**	**7.1**	**1.5**	**1.42**	**80.3**
EfficientNet-B3	224	12.0	1.0	12.5	2.0	2.71	81.6
MobileViTv2-2.0	256	18.5	7.2	8.2	3.7	11.83	81.2
EfficientViT-B2	224	24.3	1.6	12.0	2.4	36.90	82.1
FasterNet-S	224	31.1	4.5	6.3	2.2	2.41	81.3
ResNeSt50	224	27.5	5.4	13.0	2.9	32.90	81.1
ConvNeXt-T	224	29.0	4.5	5.6	3.5	68.00	82.1
Swin-T	224	29.0	4.5	8.0	2.4	14.75	81.3
PoolFormer-S36	224	31.2	5.0	13.0	5.7	3.40	81.4
EfficientFormer-L3	224	31.4	3.9	10.3	2.6	2.61	82.4
ResNet101	224	44.6	7.9	12.7	3.4	3.46	80.8
RegNetY-8G	224	39.2	8.0	11.7	3.5	3.65	81.7
ResNet152	224	60.2	4.0	19.0	6.1	4.54	81.7
PoolFormer-M36	224	56.1	8.8	13.4	7.0	5.68	82.1
FastViT-SA24	256	20.6	3.8	10.1	3.4	2.84	82.6
Next-ViT-S	224	31.7	5.8	13.5	2.8	2.90	82.5
FMViT-B	**224**	**24.3**	**4.2**	**9.1**	**2.4**	**2.40**	**82.4**
EfficientNet-B5	224	30.0	2.4	20.1	4.5	4.66	83.6
PoolFormer-M48	224	73.0	11.6	17.4	9.2	7.21	82.5
RegNetY-16G	224	83.6	16.0	13.4	5.9	6.67	82.9
EfficientFormer-L7	224	82.2	10.2	15.4	5.1	6.49	83.3
FasterNet-M	224	53.5	8.7	9.4	3.7	4.28	83.0
ResNeSt101	224	48.0	10.2	25.0	5.7	42.20	83.0
ConvNeXt-S	224	50.0	8.7	10.1	6.5	147.30	83.1
Swin-S	224	50.0	8.7	15.3	4.3	20.63	83.0
Next-ViT-B	224	44.8	8.3	20.5	4.0	3.75	83.2
FMViT-L	**224**	**35.3**	**7.1**	**15.2**	**3.9**	**3.30**	**83.3**

4.2 Object Detection and Instance Segmentation

4.2.1 Implementation

We evaluate FMViT on object detection and instance segmentation tasks based on the Mask R-CNN [9] architecture and COCO2017 [21]. Specifically, all our models are initially trained on ImageNet-1K and subsequently fine-tuned using the settings from previous work [22]. The AdamW optimizer is employed with a weight decay of 0.05, and the training spans 12 epochs. A warm-up of 500 iterations is performed during training, and the learning rate is decreased by 10 at the 8th and 11th epochs. Input resolution is 1344x800. For an equitable comparison, we solely assess backbone latency, and the testing environment aligns with that of classification.

4.2.2 Comparison with State-of-the-art Models

Table 3 presents the evaluation results utilizing the Mask R-CNN architecture. For fairness, we exclusively measured the backbone latency. As per Table 3, FMViT-B surpasses ResNet101 3.7 AP^b while achieving a 16% faster inference on TensorRT. FMViT-B matches the inference speed of PoolFormer-S12 [37], on TensorRT but with a 6.8 AP^b enhancement. Compared to EfficientFormer-L3, FMViT-B exhibits a 7% faster inference on TensorRT and a 2.7 AP^b superior performance. Against Next-ViT-S [17], FMViT-B demonstrates a 3.9 times faster inference on CoreML and a 0.3 AP^b increased performance. FMViT-L outperforms EfficientFormer-L7 by 3.8 AP^b, and its inference is 32% quicker on TensorRT. FMViT-L and ResNeSt101 have identical inference speeds on TensorRT, but FMViT-L shows a 1.2 AP^b higher performance. The AP for masks exhibits a similar advantage. In conclusion, FMViT excels in object detection and instance segmentation while maintaining a reduced inference latency.

Table 3. Comparison of different backbones on Mask R-CNN-based object detection and instance segmentation tasks. The superscripts b and m denote box detection and mask instance segmentation.

backbone	Image Size	Param (M)	FLOPs (G)	Latency(ms) pytorch	TRT	CoreML	Mask R-CNN AP^b	AP^b_{50}	AP^b_{75}	AP^m	AP^m_{50}	AP^m_{75}
ResNet101	1344x800	44.5	167.8	34.0	26.9	66.2	40.4	61.1	44.2	36.4	57.7	38.8
ResNeXt101-32x4d	1344x800	44.2	171.7	40.5	35.1	69.2	41.9	62.5	45.9	37.5	59.4	40.2
ResNeSt50	1344x800	27.5	115.6	35.9	24.1	/	42.6	/	/	38.1	/	/
Swin-T	1344x800	47.8	264.0	/	/	/	42.2	64.4	46.2	39.1	64.6	42.0
PoolFormer-S12	1344x800	11.9	39.0	48.3	23.1	32.1	37.3	59.0	40.1	34.6	55.8	36.9
PoolFormer-S24	1344x800	21.4	73.1	95.8	45.6	58.8	40.1	62.2	43.4	37.0	59.1	39.6
EfficientFormer-L3	1344x800	31.4	89.6	30.5	24.7	66.1	41.4	63.9	44.7	38.1	61.0	40.4
Next-ViT-S	1344x800	51.8	301.1	45.5	34.7	843.0	43.8	65.7	47.9	39.8	63.0	42.6
FMViT-B	**1344x800**	**72.1**	**269.9**	**27.8**	**23.1**	**213.8**	**44.1**	**66.1**	**48.0**	**41.8**	**65.1**	**45.0**
ResNeXt101-64x4d	1344x800	83.5	332.5	62.8	65.2	107.8	42.8	63.8	47.3	38.4	60.6	41.3
ResNeSt101	1344x800	48.3	219.4	63.2	42.1	/	45.2	/	/	40.2	/	/
Swin-S	1344x800	69.1	354.0	/	/	/	44.8	66.6	48.9	40.9	63.4	44.2
PoolFormer-S36	1344x800	30.9	107.2	143.3	68.9	89.2	41.0	63.1	44.8	37.7	60.1	40.0
EfficientFormer-L7	1344x800	82.2	228.6	66.9	55.6	155.5	42.6	65.1	46.1	39.0	62.2	41.7
Next-ViT-B	1344x800	64.9	354.1	61.8	48.1	932.0	45.3	67.0	49.7	41.0	64.2	44.2
FMViT-L	**1344x800**	**94.9**	**333.0**	**45.4**	**42.0**	**276.0**	**46.4**	**68.1**	**51.2**	**41.9**	**65.4**	**45.1**

4.3 ADE20K Semantic Segmentation

4.3.1 Implementation

We conducted semantic segmentation tests utilizing the ADE20K [41] dataset, which comprises approximately 20K training images and 2K validation images across 150 categories. For a fair comparison, we adhered to the training protocol of the preceding vision transformer on the Semantic FPN [15] framework. The model was initially pre-trained on ImageNet-1K at a resolution of 224x224, then trained on ADE20K with an input size of 512x512. For the Semantic FPN framework, we employed the AdamW optimizer with a learning rate and weight decay of 0.0001. We trained the entire network for 40K iterations with a total batch size of 32. Given the complexity of implementing various modules of Mask R-CNN on TensorRT and CoreML, we limited our latency assessment to the backbone for a fair comparison, maintaining the same test setup as for classification. For simplicity, We used an input size of 512x512 to measure latency.

4.3.2 Comparison with State-of-the-art Models

Table 4 illustrates that FMViT-B surpasses ResNet101 [8] by 4.7 mIoU while maintaining consistent inference speed on TensorRT

and CoreML. It exceeds Swin-T[22] by 2.0 mIoU. Compared to PoolFormer-S24 [37], it achieves 3.2 mIoU higher performance and is 8% faster in TensorRT inference. Our performance improvement of 0.5 mIoU is accompanied by 18% and 43% faster inference on TensorRT and CoreML, respectively, compared to Next-ViT-S. FMViT-L outperforms Swin-S by 1.7 mIoU and is 4.5 mIoU higher than CoreML while being 25 times faster than ResNeSt101 [39]. It matches the inference performance of PoolFormer-S36 but with a 4.9 mIoU advantage. Inference on TensorRT and CoreML is 2.5% and 29% faster than Next-ViT-B, with comparable mIoU. Comprehensive testing indicates that our FMViT holds significant potential for segmentation tasks.

Table 4. Comparison of different backbones on the ADE20K semantic segmentation task.

backbone	Image Size	Latency(ms)			Semantic FPN 80k		
		pytorch	TRT	CoreML	Param(M)	FLOPs(G)	mIoU
ResNet101	512x512	12.9	7.4	10.9	44.5	40.9	38.8
ResNeXt101-32x4d	512x512	13.8	10.1	14.1	44.2	41.8	39.7
ResNeSt50	512x512	12.7	6.5	356.0	27.5	28.2	39.7
Swin-T	512x512	/	/	/	31.9	182.0	41.5
PoolFormer-S12	512x512	9.1	4.1	6.2	11.9	9.5	37.2
PoolFormer-S24	512x512	17.8	7.8	10.8	21.4	17.8	40.3
EfficientFormer-L3	512x512	10.5	6.6	10.0	31.4	20.7	43.5
Next-ViT-S	512x512	14.5	8.5	15.6	36.3	52.0	43.0
FMViT-B	**512x512**	**9.0**	**7.2**	**10.9**	**24.8**	**22.2**	**43.5**
ResNeXt101-64x4d	512x512	17.9	17.4	22.5	83.5	81.1	40.2
ResNeSt101	512x512	25.1	11.3	423.0	48.3	53.5	42.4
Swin-S	512x512	/	/	/	53.2	274.0	45.2
PoolFormer-S36	512x512	26.4	11.5	15.2	30.9	26.1	42.0
PoolFormer-M36	512x512	41.5	16.5	22.1	56.1	46.0	42.4
EfficientFormer-L7	512x512	16.9	14.4	21.4	82.2	53.6	45.1
Next-ViT-B	512x512	20.8	12.2	21.1	49.3	64.9	47.1
FMViT-L	**512x512**	**15.2**	**11.9**	**16.3**	**36.5**	**37.4**	**46.9**

4.4 Ablation Study

We established a series of experiments to validate the efficiency of the FMB, gMLP, and RLMHSA modules within FMViT, as depicted in Table 5. Here, we incorporated our proposed modules into the FMViT-T0 model and adhered to the same training methodology as the original model. RLMHSA substitutes the traditional MHSA, gMLP supersedes the standard MPL, and FMB is not utilized for mixing; only the standard MHSA output features are directly fed into the MLP.

The experimental findings indicate that substituting the standard MHSA with our more streamlined RLMHSA decreases classification performance despite increasing parameters and FLOPs. When the conventional MLP module is replaced with the convolutional multi-group reparameterized gMLP, the number of parameters and FLOPs during the inference stage remain comparable, yet classification performance improves. Lastly, introducing the FMB module significantly increases the number of parameters and FLOPs, but it also boosts classification accuracy to the final level.

Table 5. Compare different modules.

FMB	gMLP	RLMHSA	MParams	FLOPs(M)	Top-1(%)
			2.61	479.24	68.63
		✓	2.25	338.05	67.34
	✓	✓	2.25	338.05	69.30
✓	✓	✓	2.52	395.30	72.90

4.5 Visualization

NextViT [17] establishes that CNN convolutional blocks favor high-frequency signals, while ViT is inclined towards low-frequency signals. Our proposed FMB simultaneously captures diverse high-frequency and low-frequency signals, thereby enabling the acquisition of richer texture information and more precise global information, enhancing the modeling capability of FMViT. To better understand FMViT, we visualize the Fourier spectrum of RLMHSA output features in FMVIT-T0 at both high and low frequencies. Within RLMHSA are five features with varying frequencies, each representing different frequency characteristics, denoted as f1–f5. Figure 4 illustrates this. The RLMHSA output feature, f1, captures the low-frequency signal, suggesting that RLMHSA excels at modeling global information. f2-f5, the outputs of various CFBs, capture different high-frequency signals. Each output corresponds to a distinct frequency, so they are proficient at handling various textures. The fusion of f1-f5 frequency features enhances the model's expressive capacity.

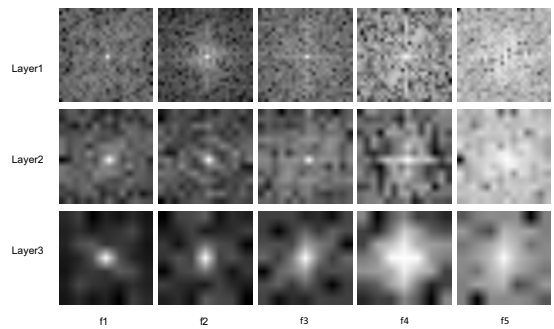

Figure 4. The Fourier spectrum of the output features of different modules in FMViT.

5 Conclusion

In this study, we introduce a hybrid ViT architecture optimized for efficient deployment on mobile devices and server GPUs. This architecture enhances the model's predictive power by amalgamating high-frequency and low-frequency features at varying frequencies, thereby bolstering the model's capacity to capture both local and global information. Experimental results demonstrate that FMViT achieves state-of-the-art latency and accuracy trade-offs across multiple vision tasks, including image classification, object detection, and semantic segmentation. However, the models we provided are stacked together without further scrutiny. Future work could employ Network Architecture Search (NAS) or other stacking methods to explore the impact of different combination models on performance.

References

[1] D. Bolya, C. Fu, X. Dai, P. Zhang, C. Feichtenhofer, and J. Hoffman. Token merging: Your vit but faster. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

[2] Y. Chen, X. Dai, D. Chen, M. Liu, X. Dong, L. Yuan, and Z. Liu. Mobile-former: Bridging mobilenet and transformer. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 5260–5269. IEEE, 2022.

[3] X. Ding, Y. Guo, G. Ding, and J. Han. Acnet: Strengthening the kernel skeletons for powerful CNN via asymmetric convolution blocks. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 1911–1920. IEEE, 2019.

[4] X. Ding, X. Zhang, J. Han, and G. Ding. Diverse branch block: Building a convolution as an inception-like unit. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 10886–10895. Computer Vision Foundation / IEEE, 2021.

[5] X. Ding, X. Zhang, J. Han, and G. Ding. Repmlp: Re-parameterizing convolutions into fully-connected layers for image recognition. *CoRR*, abs/2105.01883, 2021.

[6] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun. Repvgg: Making vgg-style convnets great again. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 13733–13742. Computer Vision Foundation / IEEE, 2021.

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[9] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2980–2988. IEEE Computer Society, 2017.

[10] Z. Hechen, W. Huang, and Y. Zhao. Vit-lsla: Vision transformer with light self-limited-attention. *CoRR*, abs/2210.17115, 2022.

[11] B. Heo, S. Yun, D. Han, S. Chun, J. Choe, and S. J. Oh. Rethinking spatial dimensions of vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 11916–11925. IEEE, 2021.

[12] G. Huang, Z. Liu, and K. Q. Weinberger. Densely connected convolutional networks. *CoRR*, abs/1608.06993, 2016.

[13] T. Huang, S. You, B. Zhang, Y. Du, F. Wang, C. Qian, and C. Xu. Dyrep: Bootstrapping training with dynamic re-parameterization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 578–587. IEEE, 2022.

[14] Z. Hui, X. Gao, Y. Yang, and X. Wang. Lightweight image super-resolution with information multi-distillation network. In *Proceedings of the 27th ACM International Conference on Multimedia, MM 2019, Nice, France, October 21-25, 2019*, pages 2024–2032. ACM, 2019.

[15] A. Kirillov, R. B. Girshick, K. He, and P. Dollár. Panoptic feature pyramid networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 6399–6408. Computer Vision Foundation / IEEE, 2019.

[16] N. Kitaev, L. Kaiser, and A. Levskaya. Reformer: The efficient transformer. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[17] J. Li, X. Xia, W. Li, H. Li, X. Wang, X. Xiao, R. Wang, M. Zheng, and X. Pan. Next-vit: Next generation vision transformer for efficient deployment in realistic industrial scenarios. *CoRR*, abs/2207.05501, 2022.

[18] W. Li, X. Wang, X. Xia, J. Wu, X. Xiao, M. Zheng, and S. Wen. Sepvit: Separable vision transformer. *CoRR*, abs/2203.15380, 2022.

[19] Y. Li, J. Hu, Y. Wen, G. Evangelidis, K. Salahi, Y. Wang, S. Tulyakov, and J. Ren. Rethinking vision transformers for mobilenet size and speed. *CoRR*, abs/2212.08059, 2022.

[20] Y. Li, G. Yuan, Y. Wen, J. Hu, G. Evangelidis, S. Tulyakov, Y. Wang, and J. Ren. Efficientformer: Vision transformers at mobilenet speed. In *NeurIPS*, 2022.

[21] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014.

[22] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 9992–10002. IEEE, 2021.

[23] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[24] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[25] N. Ma, X. Zhang, H. Zheng, and J. Sun. Shufflenet V2: practical guidelines for efficient CNN architecture design. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XIV*, volume 11218 of *Lecture Notes in Computer Science*, pages 122–138. Springer, 2018.

[26] S. Mehta and M. Rastegari. Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[27] S. Mehta and M. Rastegari. Separable self-attention for mobile vision transformers. *Trans. Mach. Learn. Res.*, 2023, 2023.

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.*, 115(3):211–252, 2015.

[29] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society, 2018.

[30] A. Srinivas, T. Lin, N. Parmar, J. Shlens, P. Abbeel, and A. Vaswani. Bottleneck transformers for visual recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 16519–16529. Computer Vision Foundation / IEEE, 2021.

[31] M. Tan and Q. V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 2019.

[32] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR, 2021.

[33] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jegou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, volume 139, pages 10347–10357, July 2021.

[34] Z. Tu, H. Talebi, H. Zhang, F. Yang, P. Milanfar, A. C. Bovik, and Y. Li. Maxvit: Multi-axis vision transformer. In *Computer Vision - ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXIV*, volume 13684 of *Lecture Notes in Computer Science*, pages 459–479. Springer, 2022.

[35] P. K. A. Vasu, J. Gabriel, J. Zhu, O. Tuzel, and A. Ranjan. An improved one millisecond mobile backbone. *CoRR*, abs/2206.04040, 2022.

[36] P. K. A. Vasu, J. Gabriel, J. Zhu, O. Tuzel, and A. Ranjan. Fastvit: A fast hybrid vision transformer using structural reparameterization. *CoRR*, abs/2303.14189, 2023.

[37] W. Yu, M. Luo, P. Zhou, C. Si, Y. Zhou, X. Wang, J. Feng, and S. Yan. Metaformer is actually what you need for vision. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 10809–10819. IEEE, 2022.

[38] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z. Jiang, F. E. H. Tay, J. Feng, and S. Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 538–547. IEEE, 2021.

[39] H. Zhang, C. Wu, Z. Zhang, Y. Zhu, H. Lin, Z. Zhang, Y. Sun, T. He, J. Mueller, R. Manmatha, M. Li, and A. J. Smola. Resnest: Split-attention networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2022, New Orleans, LA, USA, June 19-20, 2022*, pages 2735–2745. IEEE, 2022.

[40] X. Zhang, X. Zhou, M. Lin, and J. Sun. Shufflenet: An extremely efficient convolutional neural network for mobile devices. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 6848–6856. Computer Vision Foundation / IEEE Computer Society, 2018.

[41] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba. Scene parsing through ADE20K dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5122–5130. IEEE Computer Society, 2017.