Adaptive Frequency Filters As Efficient Global Token Mixers

Zhipeng Huang\textsuperscript{1,2,*} Zhizheng Zhang\textsuperscript{2†} Cuiling Lan\textsuperscript{2} Zheng-Jun Zha\textsuperscript{1} Yan Lu\textsuperscript{2} Baining Guo\textsuperscript{2} \\
\textsuperscript{1}University of Science and Technology of China  \textsuperscript{2}Microsoft Research Asia

Abstract

Recent vision transformers, large-kernel CNNs and MLPs have attained remarkable successes in broad vision tasks thanks to their effective information fusion in the global scope. However, their efficient deployments, especially on mobile devices, still suffer from noteworthy challenges due to the heavy computational costs of self-attention mechanisms, large kernels, or fully connected layers. In this work, we apply conventional convolution theorem to deep learning for addressing this and reveal that adaptive frequency filters can serve as efficient global token mixers. With this insight, we propose Adaptive Frequency Filtering (AFF) token mixer. This neural operator transfers a latent representation to the frequency domain via a Fourier transform and performs semantic-adaptive frequency filtering via an elementwise multiplication, which mathematically equals to a token mixing operation in the original latent space with a dynamic convolution kernel as large as the spatial resolution of this latent representation. We take AFF token mixers as primary neural operators to build a lightweight neural network, dubbed AFFNet. Extensive experiments demonstrate the effectiveness of our proposed AFF token mixer and show that AFFNet achieve superior accuracy and efficiency trade-offs compared to other lightweight network designs on broad visual tasks, including visual recognition and dense prediction tasks. Code is available at https://github.com/microsoft/TokenMixers.

1. Introduction

Remarkable progress has been made in ever-changing vision network designs to date, wherein effective token mixing in the global scope is constantly highlighted. Three existing dominant network families, i.e., Transformers, CNNs and MLPs achieve global token mixing with their respective ways. Transformers\textsuperscript{[16, 63, 40, 12, 80, 74]} mix tokens with self-attention mechanisms where pairwise correlations between query-key pairs are taken as mixing weights. CNNs achieve competitive performance with transformers by scaling up their kernel sizes\textsuperscript{[51, 15, 38, 9]}. MLPs\textsuperscript{[60, 25, 35]} provide another powerful paradigm via fully connections across all tokens. All of them are effective but computationally expensive, imposing remarkable challenges in practical deployments, especially on edge devices.

Recently, there is increased attention on improving the efficiency of token mixing in transformers. Some works\textsuperscript{[30, 40, 12, 22, 45, 50, 31, 75]} squeeze the scope of token mixing in different ways to compromise the representation capacities of neural networks for their efficiencies. Other works reduce the complexity of the matrix operations in self-attention by making use of the associativity property of matrix products\textsuperscript{[29]} or low-rank approximation methods\textsuperscript{[20, 76]}. These methods all sacrifice the expressiveness of neural networks and lead to unsatisfactory performance of efficient network designs. A general-purpose global token mixing for lightweight networks is still less explored. Better trade-off between accuracy and efficiency for global-scope token mixing is worthy of further study.

In this work, we reveal that adaptive frequency filters can serve as efficient global token mixers, inspired by the convolution theorem\textsuperscript{[44, 53, 48]} widely used in conventional signal processing. This theorem states that a convolution in one domain mathematically equals the Hadamard product (also known as elementwise product) in its corresponding Fourier domain. This equivalence allows us to frame...
global token mixing as a large-kernel convolution in the latent space and efficiently implement this convolution with a Hadamard product operation in the frequency domain by performing Fourier transforms on tokens in the latent space.

Besides large scopes, the adaptability to semantics also matters for token mixing as studied in [13, 8, 70, 1, 72]. This means that the weights for token mixing should be instance-adaptive. Moreover, different semantic attributes of the learned latent representations distribute in different channels [1, 73]. This property poses requirements for channel-specific token mixing wherein the weights of token mixing vary across different channels. From the perspective of framing global adaptive token mixing as a convolution, the kernel of this convolution operation should be not only large but also spatially dynamic. However, it is well known that dynamic convolutions are computationally expensive in common. Large-kernel dynamic convolutions seem extremely prohibitive for efficient/lightweight network designs. In this paper, we propose to adopt frequency filtering in the Fourier domain with learned instance-adaptive masks as a mathematical equivalent of token mixing using large-kernel dynamic convolutions by making use of the aforementioned convolution theorem. This equivalent could reduce the complexity of token mixing from $O(N^2)$ to $O(N \log N)$ thanks to adopting Fast Fourier Transforms (FFT), which is more computationally efficient.

With the key insight above, we propose Adaptive Frequency Filtering (AFF) token mixer. In this neural operator, the latent representations (i.e., a set of tokens) are transferred from its original latent space to a frequency space via a 2D discrete Fourier transform applied spatially. In this way, we get the frequency representations whose spatial positions correspond to different frequency components. We adopt an extremely lightweight network to learn instance-adaptive masks from these frequency representations, and then calculate the Hadamard product between the learned masks and the frequency representations for adaptive frequency filtering. The filtered representations are transferred back to the original latent space via an inverse Fourier transform. The features after this inverse transform could be viewed as the results of token mixing with depthwise convolution kernels whose spatial dimensions are as large as those of latent representations (i.e., the token set). According to the convolution theorem [44], our proposed operation mathematically equals to taking the tensors of applying an inverse Fourier transform to the learned masks in the Fourier domain as the corresponding kernel weights and perform convolution with this kernel in the original domain. Detailed introduction, demonstration and analysis are given in subsequent sections.

Furthermore, we take the proposed AFF token mixer as the primary neural operator and assemble it into an AFF block together with a plain channel mixer. AFF blocks serve as the basic units for constructing efficient vision backbone, dubbed AFFNet. We evaluate the effectiveness and efficiency of our proposed AFF token mixer by conducting extensive ablation study and comparison across diverse vision tasks and model scales.

Our contributions can be summarized in the following:

- We reveal that adaptive frequency filtering in the latent space can serve as efficient global token mixing with large dynamic kernels, and propose Adaptive Frequency Filtering (AFF) token mixer.
- We conduct theoretical analysis and empirical study to compare our proposed AFF token mixer with other related frequency-domain neural operators from the perspective of information fusion for figuring out what really matters for the effects of token mixing.
- We take AFF token mixer as the primary neural operator to build a lightweight vision backbone AFFNet.AFFNet achieves the state-of-the-art accuracy and efficiency trade-offs compared to other lightweight network designs across a broad range of vision tasks. An experimental evidence is provided in Fig.1.

2. Related Work

2.1. Token Mixing in Deep Learning

Mainstream neural network families, i.e., CNNs, Transformers, MLPs, differ in their ways of token mixing, as detailed in [71]. CNNs [49, 87] mix tokens with the learned weights of convolution kernels where the spatial kernel size determines the mixing scope. Commonly, the weights are deterministic and the scope is commonly a local one. Transformers [66, 16] mix tokens with pairwise correlations between query and key tokens in a local [40, 12] or global [16, 63] range. These weights are semantic-adaptive but computationally expensive due to the $O(N^2)$ complexity. MLPs commonly mix tokens with deterministic weights in manually designed scopes [5, 62, 61, 82] wherein the weights are the network parameters. This work aims to design a generally applicable token mixer for lightweight neural networks with three merits: computation-efficient, semantic-adaptive and effective in the global scope.

2.2. Lightweight Neural Networks

Lightweight neural network designs have been of high values for practical deployments. CNNs, Transformers, and MLPs have their own efficient designs. MobileNets series [27, 56, 26] introduce depthwise and pointwise convolutions as well as modified architectures for improving the efficiency. Shufflenet series [85, 41] further improve pointwise convolution via shuffle operations. MobileViT [45] combines lightweight MobileNet block and multi-head self-attention blocks. Its follow-up versions further improve it with a linear-complexity self-attention method [46].

6050
Besides, there are many works reducing the complexity of self-attention via reducing the region of token mixing [40, 12, 50, 31] or various mathematical approximations [20, 76, 42]. Many efficient MLPs limit the scope of token mixing to horizontal and vertical stripes [83, 25, 59] or a manually designed region [6].

2.3. Frequency-domain Deep Learning

Frequency-domain analysis has been a classical tool for conventional signal processing [2, 52] for a long time. Recently, frequency-domain methods begin to be introduced to the field of deep learning for analyzing the optimization [78, 79] and generalization [67, 77] capabilities of Deep Neural Networks (DNNs). Besides these, frequency-domain methods are integrated into DNNs to learn non-local [11, 54, 34, 19] or domain-generalizable [36] representations. Our proposed method might be similar to them at first glance but actually differs from them in both modelling perspectives and architecture designs. These five works propose different frequency-domain operations by introducing convolutions [11], elementwise multiplication with trainable network parameters [34], matrix multiplication with trainable parameters [34], groupwise MLP layers [19] and elementwise multiplication with spatial instance-adaptive masks [36] to frequency-domain representations, respectively. All of them are not designed for the same purpose with ours. We provide detailed mathematical analysis on their shortcomings as token mixers and conduct extensive experimental comparisons in the following sections.

3. Method

We first describe a unified formulation of token mixing, then introduce our proposed Adaptive Frequency Filtering (AFF) token mixer. We further analyze what properties matter for a frequency-domain operation in terms of its effects on token mixing. We finally introduce AFFNet which is a lightweight backbone with AFF token mixer as its core.

3.1. Unified Formulation of Token Mixing

Token mixing is of high importance since learning non-local representations is critical for visual understanding [69, 16, 63]. In most mainstream neural networks, the input image is firstly patchified into a feature tensor $X \in \mathbb{R}^{H \times W \times C}$ whose spatial resolution is $H \times W$ and the number of channels is $C$. This feature tensor could be viewed as a set of tokens, in which each token can be denoted as $x \in \mathbb{R}^{1 \times 1 \times C}$. The updated token for a query $x^q$ after token mixing in its contextual region $N(x^q)$ can be formulated in a unified form:

$$\hat{x}^q = \sum_{i \in N(x^q)} \omega^{i \rightarrow q} \times \phi(x^i), \quad (1)$$

where $\hat{x}^q$ refers to the updated $x^q$ and $x^i$ refers to the tokens in $N(x^q)$. $\phi(\cdot)$ denotes the embedding functions. $\omega^{i \rightarrow q}$ represents the weights of information fusion from token $x^i$ to the updated $x^q$. The symbol $\times$ could be Hadamard product or matrix multiplication.

We revisit the prevailing token mixing methods in different types of network architectures in terms of their effectiveness and efficiency. For CNNs, tokens are mixed by matrix multiplication with deterministic network parameters as the mixing weights. Here, the kernel sizes of convolutions determine the scopes of token mixing. This makes mixing in a global scope quite costly due to the quadratically increased parameters and FLOPs as the kernel size increases. Transformers mix tokens with pairwise correlations between query and key tokens. Its computational complexity is $O(N^2)$ ($N$ is the total number of tokens), limiting its applications in lightweight networks. Like CNNs, MLPs also mix tokens with deterministic network parameters. The scope of token mixing in advanced MLPs [5, 62, 61, 82] are commonly manually design, where the globality comes at the cost of huge computational complexity. They are all not specifically designed for lightweight neural networks.

This work aims to design a computationally efficient, semantically adaptive and global-scope token mixer for lightweight networks. This requires a large $N(x^q)$ and instance-adaptive $\omega^{i \rightarrow q}$ with less network parameters and low computation costs as possible.

3.2. Adaptive Frequency Filtering Token Mixer

We apply the convolution theorem [44, 53, 48] to deep learning for designing a token mixer with aforementioned merits for lightweight neural networks. Based on this theorem, we reveal that adaptive frequency filters can serve as efficient global token mixers. In the following, we introduce its mathematical modelling, architecture design and the equivalence between them for our proposed token mixer.

Modelling. To simplify understanding, we frame token mixing in the form of global convolution, succinctly denoted by $\hat{X} = K \ast X$. For the query token at position $(h, w)$, i.e., $X(h, w)$, Eq.(1) can be reformulated as:

$$\hat{X}(h, w) = \sum_{h' = -\lfloor \frac{H}{2} \rfloor}^{\lfloor \frac{H}{2} \rfloor} \sum_{w' = -\lfloor \frac{W}{2} \rfloor}^{\lfloor \frac{W}{2} \rfloor} K(h', w')X(h - h', w - w'), \quad (2)$$

where $\hat{X}(h, w)$ represents the updated token for $X(h, w)$ after token mixing. $H$ and $W$ are the height and weight of the input tensor, respectively. $K(h', w')$ denotes the weights for token mixing, implemented by a global convolution kernel which has the same spatial size with $X$. The padding operation for $X$ is omitted here for simplicity and the specific padding method is introduced in the subsequent parts.
With the expectation for our proposed token mixer as a semantic-adaptive and global-scope one, the weights $K$ for token mixing should be adaptive to $X$ and of a large spatial size. As illustrated by the bottom-right subfigure in Fig.2, a straightforward way for enabling $K$ adaptive to $X$ is to use a dynamic convolution kernel \( [28, 8, 23, 84] \), i.e., inferring weights of $K$ with $X$ as the inputs of a sub-network. However, adopting dynamic convolutions is usually computationally costly, even more so, when using large-kernel ones. This thus imposes big challenges in designing an efficient token mixer for lightweight networks along this way. Next, we introduce an efficient method as its equivalent implementation by making use of the convolution theorem \([44]\).

**Architecture.** The convolution theorem \([44, 48, 53]\) for inverse Fourier transform states that a convolution in one domain mathematically equals the Hadamard product in its corresponding Fourier domain. This inspires us to propose a lightweight and fast architecture (illustrated by the lower left part of Fig.2) as an extremely efficient implementation of our modelling above.

Given feature $X \in \mathbb{R}^{H \times W \times C}$, i.e., a set of tokens in the latent space, we adopt Fast Fourier Transform (FFT) to obtain the corresponding frequency representations $X_F$ by $X_F = \mathcal{F}(X)$. The detailed formulation of $\mathcal{F}(\cdot)$ is:

$$X_F(u, v) = \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} X(h, w)e^{-2\pi i(uh + vw)}.$$  \(3\)

As indicated by Eq.(3), features of different spatial positions in $X_F$ correspond to different frequency components of $X$. They incorporate global information from $X$ with a transform of $\mathcal{O}(N \log N)$ complexity.

We apply the aforementioned convolution theorem to achieve efficient global token mixing for $X$ by filtering its frequency representation $X_F$ with a learnable instance-adaptive mask. We further adopt inverse FFT to the filtered $X_F$ for getting the updated feature representations $\hat{X}$ in the original latent space. This process can be formulated as:

$$\hat{X} = \mathcal{F}^{-1}[M(\mathcal{F}(X)) \odot \mathcal{F}(X)],$$  \(4\)

where $M(\mathcal{F}(X))$ is the mask tensor learned from $X_F$, which has the same shape with $X_F$. As shown in the lower left subfigure in Fig.2, to make the network lightweight as possible, $M(\cdot)$ is efficiently implemented by a group $1 \times 1$ convolution (linear) layer, followed by a ReLU function and another group linear layer. $\odot$ denotes Hadamard product, also known as elementwise multiplication, and $\mathcal{F}^{-1}(\cdot)$ denotes inverse Fourier transform. Here, $\hat{X}$ can be viewed as the results of global adaptive token mixing for $X$, which is mathematically equivalent to adopting a large-size dynamic convolution kernel as the weights for token mixing. The equivalence is introduced in the following.

**Equivalence.** The convolution theorem still applies to the latent representations of neural networks. The multiplication of two signals in the Fourier domain equals to the
Fourier transform of a convolution of these two signals in their original domain. When applying this to the frequency-domain multiplication in Fig.(2), we know that:

$$\mathcal{M}(\mathcal{F}(X)) \circ \mathcal{F}(X) = \mathcal{F}(\mathcal{F}^{-1}[\mathcal{M}(\mathcal{F}(X))] * X).$$  \hspace{1cm} (5)$$

Combining Eq.(4) and Eq.(5), it is easy to get that:

$$\hat{X} = \mathcal{F}^{-1}[\mathcal{M}(\mathcal{F}(X))] * X,$$  \hspace{1cm} (6)$$

where $\mathcal{F}^{-1}(\mathcal{M}(\mathcal{F}(X)))$ is a tensor of the same shape with $X$, which could be viewed as a dynamic depthwise convolution kernel as large as $X$ in spatial. This kernel is adaptive to the contents of $\hat{X}$. Due to the property of Fourier transform [44], a circular padding is adopted to $X$ here as shown in Fig.2. So far, we understand why the operation in Eq.(4) mathematically equals to a global-scope token mixing operation with semantic-adaptive weights.

3.3. Analysis

As introduced in Sec.2.3, there have been some studies applying frequency-domain methods to DNN for learning non-local or domain-generalizable representations in previous works [11, 54, 34, 19, 36]. They are all designed for different purposes with ours. In this section, we revisit the frequency-domain operations in these works from the perspective of token mixing and compare our design with them.

FFC [11] and AFNO [19] adopt linear (also known as $1 \times 1$ convolution) layers with non-linear activation functions to the representations in the frequency domain. Specifically, AFNO [19] adopt a linear layer followed by a ReLU function, another linear layer and a SoftShrink\(^1\) function to the frequency representations after Fourier transforms, which can be briefly described as $FFT \rightarrow Linear \rightarrow ReLU \rightarrow Linear \rightarrow SoftShrink \rightarrow iFFT$. Here, linear layer and Fourier transform are in fact commutative, i.e., $\mathcal{F}(\mathcal{F}(X)) = \mathcal{F}(\text{Linear}(X))$, which can be proved with the distributive property of matrix multiplication by:

$$W_{Linear} \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} X(h, w) e^{-2\pi i (uh + vw)}$$

$$= \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} (W_{Linear} X(h, w)) e^{-2\pi i (uh + vw)},$$  \hspace{1cm} (7)$$

where $W_{Linear}$ denotes the parameters of a linear layer. We know that successive Fourier transform and its inverse transform equal to an identity function. Thus, the architecture of AFNO could be rewrote as: $FFT \rightarrow Linear \rightarrow ReLU \rightarrow (iFFT \rightarrow FFT) \rightarrow Linear \rightarrow SoftShrink \rightarrow iFFT$. Upon the commutative law proved in Eq.(7), we can know this architecture is in fact equivalent to $Linear \rightarrow FFT \rightarrow ReLU \rightarrow iFFT$.

\(^1\)https://pytorch.org/docs/stable/generated/torch.nn.Softshrink.html

<table>
<thead>
<tr>
<th>Properties</th>
<th>FFC</th>
<th>AFNO</th>
<th>GFNet</th>
<th>FNO</th>
<th>DFF</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic-adaptive</td>
<td>✓</td>
<td>☒</td>
<td>☒</td>
<td>☒</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Variable-size input</td>
<td>✓</td>
<td>✓</td>
<td>☒</td>
<td>☒</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Channel-wise mix</td>
<td>✓</td>
<td>✓</td>
<td>☒</td>
<td>☒</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 1. Comparisons of our proposed AFF token mixer with other frequency-domain neural operators in terms of three important properties for token mixing.

$→Linear→FFT→SoftShrink→iFFT$. Now, it is easy to find that only ReLU and SoftShrink functions remain in the Fourier domain. These two deterministic functions cannot achieve semantic-adaptive filtering as our proposed AFF token mixer does. The same problem also exists in FFC [11].

GFNet [54] and FNO [34] multiply the representations after Fourier transforms with trainable network parameters. GFNet [54] adopts elementwise multiplication while FNO [34] uses matrix multiplication. Both of them are not semantic-adaptive since the masks implemented by network parameters are shared over different instances and fixed after training. Besides, they cannot support for variable-size inputs since the shapes of these masks are fixed, leading to the lack of flexibility in their practical using.

DFF [36] learns a spatial mask to filter out frequency components that are not conductive to domain generalization. It is proposed for domain generalization problems in which only spatial mask is needed as studied in [36] since different spatial position of the features after a Fourier transform correspond to different frequency components. However, it is not competent as a token mixer since the learned mask is shared along the channel dimension. This means that the weights for its equivalent token mixing are shared for different channels. However, different channels commonly represent different semantic attributes [1, 73], thus requiring adaptive weights in token mixing.

We summarize the comparisons of different frequency-domain designs in terms of three important properties for token mixing in Table 1. The results of experimental verification are in Table 5 as follows.

3.4. Network Architectures

With our AFF token mixer as the core neural operator, we introduce its corresponding module and network design.

**AFF Block**  For the output $X^{l-1}$ of the $(l-1)$-th AFF Block, we adopt the commonly used module MBConv [46, 57, 58, 45, 64] with Layer Normalization (LN) for channel mixing, then feed it to our proposed AFF token mixer for global token mixing to get the output of $l$-th AFF block. Skip-connections for channel mixing and token mixing are adopted to facilitate model training. The entire architecture of AFF Block can be formulated as:

$$ affine(x) = \text{Linear}(x) \circ \text{ReLU}(x) \circ \text{SoftShrink}(x) \circ iFFT(x)$$
### AFFNet

We stack multiple AFF blocks for constructing a lightweight backbone network, namely AFFNet, as shown in Fig. 2. Following the common practices [45, 46], we employ a convolution stem for tokenization and a plain fusion for combining local and global features at each stage. We replace the standard convolutions in the detection head for backbone networks with comparable model sizes in Top-1 accuracy. The AFFNet reaches 79.8% Top-1 accuracy with 5.5M parameters and 1.5G FLOPs. Our extremely tiny model AFFNet-ET attains 73% Top-1 accuracy with 5.5M parameters and 1.5G FLOPs. Our extremely tiny AFFNet-T and extremely tiny AFFNet-ET versions have 5.5M, 2.6M and 1.4M parameters, respectively. Their detailed configurations are in the Supplementary.

### 4. Experiments

We evaluate our proposed AFF token mixer by conducting comparisons with the state-of-the-art lightweight networks and extensive ablation studies for its design.

#### 4.1. Image Classification

**Settings.** We train different versions of our proposed lightweight networks AFFNet as backbones on ImageNet-1k dataset [55] from scratch. All models are trained for 300 epochs on 8 NVIDIA V100 GPUs with a batch size of 1024. More implementation details are in the Supplementary.

**Results.** We report the comparison results between our proposed AFFNet and other SOTA lightweight models in Table 2. We observe that our AFFNet outperforms other lightweight networks with comparable model sizes in Top-1 accuracy. The AFFNet reaches 79.8% Top-1 accuracy with 5.5M parameters and 1.5G FLOPs. Our extremely tiny model AFFNet-ET attains 73% Top-1 accuracy with solely 1.4M and 0.4G FLOPs. As a result, AFFNet achieves the best trade-offs between accuracy and efficiency. To show the comparison results more intuitively, we illustrate the accuracy and efficiency trade-offs of our AFFNet and some advanced lightweight models with global token mixers in Fig. 1. Thanks to AFF token mixer, AFFNet is superior to them by a clear margin across different model scales. Its superiority is especially significant when the model is extremely tiny, which demonstrates the effectiveness of AFF token mixer on information fusion at very low costs. AFFNet, AFFNet-T, and AFFNet-ET models achieve 4202, 5304, and 7470 images/s throughput on ImageNet-1K tested with one NVIDIA A100 GPU, respectively, which is 13.5%, 8.2%, and 14.9% faster than MobileViT-S/XS/XXS. More detailed results are in the Supplementary.

<table>
<thead>
<tr>
<th>Model</th>
<th>Pub.</th>
<th>Res.</th>
<th>Param. (M)</th>
<th>FLOPs (G)</th>
<th>Top-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNetv1-0.5 [27]</td>
<td>arXiv17</td>
<td>224^2</td>
<td>1.3</td>
<td>0.2</td>
<td>63.7</td>
</tr>
<tr>
<td>MViT-XXS [45]</td>
<td>ICLR22</td>
<td>256^2</td>
<td>1.3</td>
<td>0.3</td>
<td>71.2</td>
</tr>
<tr>
<td>EdgeNeilX-XXS [43]</td>
<td>ECCV22</td>
<td>256^2</td>
<td>1.3</td>
<td>0.3</td>
<td>71.2</td>
</tr>
<tr>
<td>MViTv2-0.5 [56]</td>
<td>TMLR23</td>
<td>256^2</td>
<td>1.3</td>
<td>0.3</td>
<td>71.2</td>
</tr>
<tr>
<td>AFFNet-ET</td>
<td>-</td>
<td>256^2</td>
<td>1.3</td>
<td>0.3</td>
<td>71.2</td>
</tr>
<tr>
<td>MNetv3-L-0.5 [26]</td>
<td>ICCV19</td>
<td>224^2</td>
<td>2.6</td>
<td>0.1</td>
<td>68.8</td>
</tr>
<tr>
<td>MFormer-S2 [7]</td>
<td>CVPR22</td>
<td>224^2</td>
<td>3.6</td>
<td>0.1</td>
<td>68.7</td>
</tr>
<tr>
<td>PVTv2-B0 [63]</td>
<td>CVM22</td>
<td>224^2</td>
<td>3.7</td>
<td>0.6</td>
<td>70.5</td>
</tr>
<tr>
<td>MViT-XS [45]</td>
<td>ICLR22</td>
<td>256^2</td>
<td>2.3</td>
<td>1.0</td>
<td>74.8</td>
</tr>
<tr>
<td>EdgeNeil-XS [43]</td>
<td>ECCV22</td>
<td>256^2</td>
<td>2.3</td>
<td>0.5</td>
<td>75.0</td>
</tr>
<tr>
<td>EFormer-S0 [32]</td>
<td>arXiv22</td>
<td>224^2</td>
<td>3.5</td>
<td>0.4</td>
<td>75.7</td>
</tr>
<tr>
<td>MViTv2-0.75 [46]</td>
<td>TMLR23</td>
<td>256^2</td>
<td>2.9</td>
<td>1.0</td>
<td>75.6</td>
</tr>
<tr>
<td>AFFNet-T</td>
<td>-</td>
<td>256^2</td>
<td>2.6</td>
<td>0.8</td>
<td>77.0</td>
</tr>
<tr>
<td>MNetv2 [56]</td>
<td>CVPR18</td>
<td>224^2</td>
<td>6.9</td>
<td>0.6</td>
<td>74.7</td>
</tr>
<tr>
<td>ShuffleNetV2 [41]</td>
<td>ECCV18</td>
<td>224^2</td>
<td>5.5</td>
<td>0.6</td>
<td>74.5</td>
</tr>
<tr>
<td>MNetv3 [26]</td>
<td>ICCV19</td>
<td>224^2</td>
<td>5.4</td>
<td>0.2</td>
<td>75.2</td>
</tr>
<tr>
<td>T2T-ViT [81]</td>
<td>ICCV21</td>
<td>224^2</td>
<td>6.9</td>
<td>1.8</td>
<td>76.5</td>
</tr>
<tr>
<td>DetF-T [63]</td>
<td>ICLM21</td>
<td>224^2</td>
<td>5.7</td>
<td>1.3</td>
<td>72.2</td>
</tr>
<tr>
<td>CoatF-Lite-T [14]</td>
<td>ICCV21</td>
<td>224^2</td>
<td>5.7</td>
<td>1.6</td>
<td>77.5</td>
</tr>
<tr>
<td>LeViT-128 [18]</td>
<td>ICCV21</td>
<td>224^2</td>
<td>9.2</td>
<td>0.4</td>
<td>78.6</td>
</tr>
<tr>
<td>GFNet-Ti [54]</td>
<td>NeurIPS21</td>
<td>224^2</td>
<td>7.0</td>
<td>1.3</td>
<td>74.6</td>
</tr>
<tr>
<td>EFormer-L1 [33]</td>
<td>NeurIPS22</td>
<td>224^2</td>
<td>12.3</td>
<td>1.3</td>
<td>79.2</td>
</tr>
<tr>
<td>EFormer-S1 [32]</td>
<td>arXiv22</td>
<td>224^2</td>
<td>6.1</td>
<td>0.7</td>
<td>79.0</td>
</tr>
<tr>
<td>MFormer [7]</td>
<td>CVPR22</td>
<td>224^2</td>
<td>9.4</td>
<td>0.2</td>
<td>76.7</td>
</tr>
<tr>
<td>EfficientViT [3]</td>
<td>arXiv22</td>
<td>224^2</td>
<td>7.9</td>
<td>0.4</td>
<td>78.6</td>
</tr>
<tr>
<td>EdgeViT-XS [10]</td>
<td>ECCV22</td>
<td>224^2</td>
<td>6.7</td>
<td>1.1</td>
<td>77.5</td>
</tr>
<tr>
<td>MOne-S3 [63]</td>
<td>arXiv22</td>
<td>224^2</td>
<td>10.1</td>
<td>1.9</td>
<td>78.1</td>
</tr>
<tr>
<td>MViT-S [45]</td>
<td>ICLR22</td>
<td>256^2</td>
<td>5.6</td>
<td>2.0</td>
<td>78.4</td>
</tr>
<tr>
<td>EdgeNext-S [43]</td>
<td>ECCV22</td>
<td>256^2</td>
<td>5.6</td>
<td>1.3</td>
<td>79.4</td>
</tr>
<tr>
<td>MViTv2-1.0 [46]</td>
<td>TMLR23</td>
<td>256^2</td>
<td>4.9</td>
<td>1.8</td>
<td>78.1</td>
</tr>
<tr>
<td>AFFNet</td>
<td>-</td>
<td>256^2</td>
<td>5.5</td>
<td>1.5</td>
<td>79.8</td>
</tr>
</tbody>
</table>

Table 2. Comparisons of our proposed AFFNet with other state-of-the-art lightweight networks on ImageNet-1K classification over different model scales (i.e., <2M, 2M ~ 4M and > 4M). For conciseness, Pub., Res., Param., MNet, MOne, EFormer and MViT are short for Publication, Resolution, Parameters, MobileNet, MobileOne, MobileFormer, EfficientFormer and MobileViT, respectively.

### 4.2. Object Detection

**Settings.** We conduct object detection experiments on MS-COCO dataset [37]. Following the common practices in [27, 56, 45, 46, 43], we compare different lightweight backbones upon the Single Shot Detection (SSD) [39] framework wherein separable convolutions are adopted to replace the standard convolutions in the detection head for evaluation in the lightweight setting. In the training, we load ImageNet-1K pre-trained weights as the initialization of the backbone network, and fine-tune the entire model on the training set of MS-COCO with the AdamW optimizer for 200 epochs. The input resolution of the images is 320 × 320. Detailed introduction for the used dataset and more implementation details are in the Supplementary.

**Results.** As shown in Table 3, the detection models equipped with AFFNet consistently outperforms other lightweight CNNs or transformers based detectors in mAP.
Table 3. Comparisons of our AFFNet with other state-of-the-art models for object detection on COCO dataset, and segmentation on ADE20k and VOC dataset. Here, Param., MNet and MViT are short for Parameters, MobileNet and MobileViT, respectively.

<table>
<thead>
<tr>
<th>Model</th>
<th>Detection</th>
<th>Segmentation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Param. mAP (%) COCO</td>
<td>Param. mIOU (%) ADE20kVOC</td>
</tr>
<tr>
<td>MViT-XS [45]</td>
<td>1.9 18.5</td>
<td>1.9 73.6</td>
</tr>
<tr>
<td>MViTv2-0.5 [46]</td>
<td>2.0 21.2</td>
<td>3.6 31.2 75.1</td>
</tr>
<tr>
<td>AFFNet-ET</td>
<td>1.9 21.8</td>
<td>2.2 33.0 76.1</td>
</tr>
<tr>
<td>MViT-XS [45]</td>
<td>2.7 24.8</td>
<td>2.9 - 77.1</td>
</tr>
<tr>
<td>MViTv2-0.75 [46]</td>
<td>3.6 24.6</td>
<td>6.2 34.7 75.1</td>
</tr>
<tr>
<td>AFFNet-T</td>
<td>3.0 25.3</td>
<td>3.5 36.9 77.8</td>
</tr>
<tr>
<td>ResNet-50 [24]</td>
<td>22.9 25.2</td>
<td>68.2 32.6 76.8</td>
</tr>
<tr>
<td>MNetv1 [27]</td>
<td>5.1 22.2</td>
<td>11.2 - 75.3</td>
</tr>
<tr>
<td>MNetv2 [56]</td>
<td>4.3 22.1</td>
<td>18.7 34.1 75.7</td>
</tr>
<tr>
<td>MViT-S [45]</td>
<td>5.7 27.7</td>
<td>6.4 - 79.1</td>
</tr>
<tr>
<td>MViTv2-1.0 [46]</td>
<td>5.6 26.5</td>
<td>9.4 37.0 78.9</td>
</tr>
<tr>
<td>EdgeNext [43]</td>
<td>6.2 27.9</td>
<td>6.5 - 80.2</td>
</tr>
<tr>
<td>AFFNet</td>
<td>5.6 28.4</td>
<td>6.9 38.4 80.5</td>
</tr>
</tbody>
</table>

Table 4. Comparisons of our proposed model with baseline (no spatial token mixer) and models with other token mixers in the original domain on ImageNet-1K classification. “Base.” denotes the baseline model discarding all AFF token mixers. “Conv-Mixer (3×3)” refers to adopting token mixers implemented by 3×3 convolutions in the original space. “AFF w/o FFT” denotes performing adaptive filtering in the original space with the same networks by discarding the Fourier transforms where “w/o” and “AFF” are short for “without” and “AFF token mixer”, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Param (M)</th>
<th>FLOPs (G)</th>
<th>Top-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base.</td>
<td>5.2</td>
<td>1.3</td>
<td>77.9</td>
</tr>
<tr>
<td>Base. + Conv-mixer (3×3)</td>
<td>10.7</td>
<td>2.7</td>
<td>78.6</td>
</tr>
<tr>
<td>Base. + AFF w/o FFT</td>
<td>5.5</td>
<td>1.5</td>
<td>78.4</td>
</tr>
<tr>
<td>Base. + AFF (Our AFFNet)</td>
<td>5.5</td>
<td>1.5</td>
<td>79.8</td>
</tr>
</tbody>
</table>

4.4. Ablation Study

Effectiveness and complexity of AFF token mixer. We analyze the effectiveness and complexity of our proposed AFF token mixer by comparing AFFNet with the Base. model in which all AFF token mixers are replaced with identity functions. As shown in Table 4, all AFF token mixers in AFFNet only require 0.3M parameter increase (∼6%) and 0.2G FLOPs increase (∼15%) relative to the baseline and improves the Top-1 accuracy on ImageNet-1K by 1.9%. Comparing to the model with one 3×3 convolution layer as the token mixer, i.e., Base. + Conv-Mixer (3×3), AFFNet delivers 1.2% Top-1 accuracy improvements with about only half of parameters and FLOPs. This strongly demonstrates the effectiveness and efficiency of our proposed method for token mixing in lightweight networks.

Original vs. frequency domain. We compare applying the same adaptive filtering operations in original domain and in frequency domain. We discard the all Fourier and inverse Fourier transforms and remain others the same as AFFNet, i.e., Base.+AFF w/o FFT in Table 4. Our AFFNet clearly outperforms it by 1.4% Top-1 accuracy with the same model complexity. Applying adaptive filtering in the original domain is even weaker than convolutional token mixer, which indicates that only adaptive frequency filters can serve as efficient global token mixers.

Comparisons of different frequency operations. We compare the frequency operation design in AFF token mixer with those in previous works [34, 54, 11, 36, 19] in terms of their effects as token mixers. The results are in Table 5. As analyzed in Sec.3.3, FFC [11] and AFNO [19] actually perform filtering with deterministic functions, resulting in the lack of the adaptivity to semantics. The frequency-
Table 6. Experiments of verifying the importance of channel-
specific token mixing on ImageNet-1K. Here, we adopt an aver-
age pooling operation along the channel dimension of the masks
learned in AFFNet, yielding the mask with a shape of 1×H×W.
This mask is shared across channels.

<table>
<thead>
<tr>
<th>Mask Shape</th>
<th>Param (M)</th>
<th>FLOPs (G)</th>
<th>Top-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1×H×W</td>
<td>5.5</td>
<td>1.5</td>
<td>79.1</td>
</tr>
<tr>
<td>C×H×W</td>
<td>5.5</td>
<td>1.5</td>
<td>79.8</td>
</tr>
</tbody>
</table>

Table 7. Comparisons of different hyper-parameter choices in
the sub-network for learning the filtering masks in AFFNet on
ImageNet-1K. “Spatial K-Size” refers to the spatial size of con-
volution kernels. \(N_{\text{group}}\) denotes the number of groups for group
linear or convolution layers. \(C\) is the total number of channels.

<table>
<thead>
<tr>
<th>Spatial K-Size</th>
<th>(N_{\text{group}})</th>
<th>Param (M)</th>
<th>FLOPs (G)</th>
<th>Top-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 × 1</td>
<td>(C)</td>
<td>5.3</td>
<td>1.4</td>
<td>79.4</td>
</tr>
<tr>
<td>1 × 1</td>
<td>1</td>
<td>7.7</td>
<td>2.0</td>
<td>79.9</td>
</tr>
<tr>
<td>3 × 3</td>
<td>8</td>
<td>7.9</td>
<td>2.0</td>
<td>79.8</td>
</tr>
<tr>
<td>1 × 1</td>
<td>8</td>
<td>5.5</td>
<td>1.5</td>
<td>79.8</td>
</tr>
</tbody>
</table>

The importance of channel-specific token mixing. We
demonstrate this by comparing the frequency-domain oper-
ations in them are both obviously inferior to
ours. Moreover, our operation design is also clearly better
than those in GFN [54] and FNO [34] since they perform filter-
ing with network parameters implemented masks. These
masks are fixed after training and lead to a large increase in
parameters (\(\text{Base.} + \text{FNO}\) has more than 25 × parameters as
ours). Note that the implementation of FNO [34] with un-
shared fully connected layers for each frequency component
results in a significant increase in the number of parameters.
DFF [36] is designed for filtering out the frequency com-
ponents adverse to domain generalization, thus requiring a
spatial mask only. Our AFFNet is superior to \(\text{Base.} + \text{DFF}\)
by 0.5% with fewer parameters and FLOPs, demonstrating
the importance of channel-wise mixing. This will be further
verified with a fairer comparison. These existing frequency-
domain operations might be similar with our proposed one
at the first glance, but they are designed for different pur-
poses and perform worse than ours as token mixers. When
replacing the Hadamard product in our method with a sum-
mary operation, the Top-1 accuracy drops by 1.0% since
the equivalence introduced in Sec.3.2 no longer holds.

Comparisons of hyper-parameter choices. As shown in
Fig.2, we adopt two group linear layers (also known as 1 × 1
convolution layers) with ReLU to learn the masks for our
proposed adaptive frequency filtering. As shown in Table
7, improving the kernel size cannot further improve the per-
formance but leads to larger model complexities. Moreover,
we keep the spatial kernel size as 1 × 1 while using different
 group numbers. When \(N_{\text{group}}=C\), the Top-1 accuracy drops by 0.4%, in which depthwise convolutions are used
so that the contexts among different channels are under-
explotted for inferring the weights of token mixing. When
\(N_{\text{group}}=1\), it means that regular convolution/linear layers
are used, which slightly improve the Top-1 accuracy by
0.1% at the expense of 40% parameters increase and 33.3%
FLOPs increase. This setting explores more contexts but
results in a worse accuracy and efficiency trade-off.

5. Conclusion

In this work, we reveal that adaptive frequency filters
can serve as efficient global token mixers in a mathemati-
cally equivalent manner. Upon this, we propose Adaptive
Frequency Filtering (AFF) token mixer to achieve low-
cost adaptive token mixing in the global scope. More-
over, we take AFF token mixers as primary neural op-
erators to build a lightweight backbone network, dubbed
AFFNet. AFFNet achieves SOTA accuracy and efficiency
trade-offs compared to other lightweight network designs
across multiple vision tasks. Besides, we revisit the existing
frequency-domain neural operations for figuring out what
matters in their designs for token mixing. We hope this
work could inspire more interplay between conventional
signal processing and deep learning technologies.
References


[34] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations. ICLR, 2021. 3, 5, 7, 8


[38] Shihwei Liu, Tianlong Chen, Xiaohan Chen, Xuxi Chen, Qiao Xiao, Boqian Wu, Mykola Pechenizkiy, Decibel Mocanu, and Zhangyang Wang. More convnets in the 2020s: Scaling up kernels beyond 51x51 using sparsity. ICLR, 2023. 1


[40] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In ICCV, 2021. 1, 2, 3


[45] Sachin Mehta and Mohammad Rastegari. Separable self-attention for mobile vision transformers. TMLR, 2022. 2, 5, 6, 7


[63] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiang Li. Maxvit: Multi-axis vision transformer. In ECCV, 2022. 5


[68] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pvt v2: Improved baselines with pyramid vision transformer. CVM, 2022. 6
[70] Guoqiang Wei, Zhizheng Zhang, Cuiling Lan, Yan Lu, and Zhibo Chen. Active token mixer. In AAAI, 2023. 2
[71] Guoqiang Wei, Zhizheng Zhang, Cuiling Lan, Yan Lu, and Zhibo Chen. Active token mixer. AAAI, 2023. 2