MaxViT: Multi-Axis Vision Transformer (Supplementary Material)

Zhengzhong Tu[1,2], Hossein Talebi[1], Han Zhang[1], Feng Yang[1], Peyman Milanfar[1], Alan Bovik[2], and Yinxiao Li[1]

[1] Google Research
[2] University of Texas at Austin

In this manuscript we provide the following material:

- Sec. 1 describes the detailed architectures of MaxViT for image classification (Sec. 1.1), object detection and segmentation (Sec. 1.2), image aesthetics assessment (Sec. 1.3), and image generation (Sec. 1.4).
- Sec. 2 presents complete training settings and hyperparameters for image classification (Sec. 2.1), object detection and segmentation (Sec. 2.2), image aesthetics assessment (Sec. 2.3), and image generation (Sec. 2.4).
- Sec. 3 demonstrates comprehensive experimental results, including image classification on ImageNet-1K (Table 3), ImageNet-21K and JFT (Table 4), as well as more image generation visualizations on ImageNet-1K (Figure 3).

1 Model Details

1.1 Backbone Details

MBConv MaxViT leverages the MBConv block [30,34] as the main convolution operator. We also adopt a pre-activation structure [5,10] to promote homogeneity between MBConv and Transformer blocks. Specifically, assume $\mathbf{x}$ to be the input feature, the MBConv block without downsampling is formulated as:

$$\mathbf{x} \leftarrow \mathbf{x} + \mathsf{Proj}(\mathsf{SE}(\mathsf{DWConv}(\mathsf{Conv}(\mathsf{Norm}(\mathbf{x}))))), \tag{1}$$

where Norm is BatchNorm [14], Conv is the expansion Conv1x1 followed by BatchNorm and GELU [11] activation, a typical choice for Transformer-based models. DWConv is the Depthwise Conv3x3 followed by BatchNorm and GELU. SE is the Squeeze-Excitation layer [13], while Proj is the shrink Conv1x1 to down-project the number of channels. Note that for the first MBConv block in every stage, the downsampling is done by applying stride-2 Depthwise Conv3x3 while the shortcut branch should also apply pooling and channel projection:

$$\mathbf{x} \leftarrow \mathsf{Proj}(\mathsf{Pool2D}(\mathbf{x})) + \mathsf{Proj}(\mathsf{SE}(\mathsf{DWConv}{\downarrow}(\mathsf{Conv}(\mathsf{Norm}(\mathbf{x})))))). \tag{2}$$

Relative Attention Relative attention has been explored in several previous studies for both NLP [31, 40] and vision [5, 15, 23, 38]. Here to simplify the presentation, we present our model using only a single head of the multi-head

self-attention. In the actual implementation, we always use multi-head attention with the same head dimension. The relative attention can be defined as:

$$\mathsf{RelAttention}(Q, K, V) = \mathsf{softmax}(QK^T/\sqrt{d} + B)V, \tag{3}$$

where $Q, K, V \in \mathbb{R}^{(H\times W)\times C}$ are the query, key, and value matrices and d is the hidden dimension. The attention weights are co-decided by a learned static location-aware matrix B and the scaled input-adaptive attention $QK^T/\sqrt{d}$. Considering the differences in 2D coordinates, the relative position bias B is parameterized by a matrix $\hat{B} \in \mathbb{R}^{(2H-1)(2W-1)}$. Following typical practices [5,23], when fine-tuned at a higher resolution *e.g.*, $H' \times W'$, we use bilinear interpolation to map the relative positional bias from $\mathbb{R}^{(2H-1)(2W-1)}$ to $\mathbb{R}^{(2H'-1)(2W'-1)}$. This relative attention benefits from input-adaptivity, translation equivariance, and global interactions, which is a preferred choice over the vanilla self-attention on 2D vision tasks. In our model, all the attention operators use this relative attention defined in Eq. 3 by default.

Multi-Axis Attention We assume the relative attention operator in Eq. 3 follows the convention for 1D input sequences *i.e.*, always regards the *second last dimension* of an input $(..., L, C)$ as the *spatial axis* where L, C represent sequence length and channels. The proposed Multi-Axis Attention can be implemented without modification to the self-attention operation. To start with, we first define the $\mathsf{Block}(\cdot)$ operator with parameter P as partitioning the input image/feature $\mathbf{x} \in \mathbb{R}^{H\times W\times C}$ into non-overlapping blocks with each block having size $P \times P$. Note that after window partition, the block dimensions are gathered onto the spatial dimension (*i.e.*, -2 axis):

$$\mathsf{Block} : (H, W, C) \to (\frac{H}{P} \times P, \frac{W}{P} \times P, C) \to (\frac{HW}{P^2}, P^2, C). \tag{4}$$

We denote the $\mathsf{Unblock}(\cdot)$ operation as the reverse of the above block partition procedure. Similarly, we define the $\mathsf{Grid}(\cdot)$ operation with parameter G as dividing the input feature into a uniform $G \times G$ grid, with each lattice having *adaptive size* $\frac{H}{G} \times \frac{W}{G}$. Unlike the block operator, we need to apply an extra $\mathsf{Transpose}$ to place the grid dimension in the assumed spatial axis (*i.e.*, -2 axis):

$$\mathsf{Grid} : (H, W, C) \to (G \times \frac{H}{G}, G \times \frac{W}{G}, C) \to \underbrace{(G^2, \frac{HW}{G^2}, C) \to (\frac{HW}{G^2}, G^2, C)}_{\text{swapaxes(axis1=-2,axis2=-3)}} \tag{5}$$

with its inverse operation $\mathsf{Ungrid}(\cdot)$ that reverses the gridded input back to the normal 2D feature space.

To this end, we are ready to explain the multi-axis attention module. Given an input tensor $\mathbf{x} \in \mathbb{R}^{H\times W\times C}$, the local Block Attention can be expressed as:

$$\begin{aligned} \mathbf{x} &\leftarrow \mathbf{x} + \mathsf{Unblock}(\mathsf{RelAttention}(\mathsf{Block}(\mathsf{LN}(\mathbf{x})))) \\ \mathbf{x} &\leftarrow \mathbf{x} + \mathsf{MLP}(\mathsf{LN}(\mathbf{x})) \end{aligned} \tag{6}$$

while the global, dilated Grid Attention module is formulated as:

$$\begin{aligned} \mathbf{x} &\leftarrow \mathbf{x} + \mathsf{Ungrid}(\mathsf{RelAttention}(\mathsf{Grid}(\mathsf{LN}(\mathbf{x})))) \\ \mathbf{x} &\leftarrow \mathbf{x} + \mathsf{MLP}(\mathsf{LN}(\mathbf{x})) \end{aligned} \tag{7}$$

where we omit the QKV input format in the RelAttention operation for simplicity. LN denotes the Layer Normalization [1], where MLP is a standard MLP network [7, 23] consisting of two linear layers: $\mathbf{x} \leftarrow W_2\mathsf{GELU}(W_1\mathbf{x})$.

Comparison to Axial attention It should be noted that our proposed multi-axis attention (Max-SA) module is completely different from the axial attention proposed in [12, 39]. As shown in Figure 1(a), Axial attention proposes to first apply column-wise attention then row-wise, which achieves a global receptive field with $\mathcal{O}(N\sqrt{N})$ complexity (assuming N equals to the number of pixels). On the contrary, our proposed Max-SA shown in Figure 1(b) first employs local attention, then sparse global attention, enjoying global receptive fields with only $\mathcal{O}(N)$ *linear* complexity. Moreover, we deem the proposed Max-SA a more natural approach for vision since the design of attended regions account for the 2D structure of images, *e.g.*, mixing tokens in a spatially-local small window.

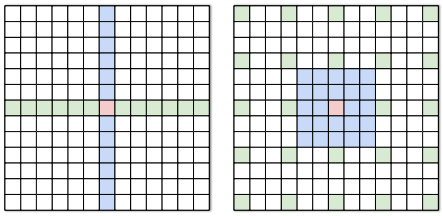

Fig. 1: Comparison of Axial attention and our proposed Multi-Axis attention.

MaxViT Block We demonstrate in Algo. 1 an `einops`-style pseudocode of the MaxViT block which contains MBConv, block attention, and grid attention.

Classification Head Instead of using the `[cls]` token [7], we simply apply global average pooling to the output of the last stage (S4) to obtain the feature representation, followed by the final classification head.

Architectural Specifications Finally, we present detailed architectural specifications for the MaxViT model family (T/S/B/L) in Table 1.

1.2 Detection and Segmentation Models

We follow the settings of the cascaded Faster-RCNN [29] and Mask-RCNN [9], but replace the feature extraction backbone with our MaxViT backbone. We also applied FPN [21] in the feature map generation, where the S2, S3, S4 (multi-scale features of targeted resolution 1/8, 1/16, 1/32 in MaxViT, respectively) are used. Then the generated feature maps are fed into the detection head. For

Algo. 1 Pseudocode of MaxViT Block

```
# input: features (b, h, w, c). Assume h==w; x/output: features (b, h, w, c).
# p/g: block/grid size. Use 7 by default.

def RelSelfAttn(x): return x # A self-attn function applied on the -2 axis

# Window/grid partition function
from einops import rearrange
def block(x,p):
  return rearrange(x,"b(hy)(wx)c->b(hw)(yx)c",h=x.shape[1]//p,w=x.shape[2]//p,y=p,x=p)

def unblock(x,g,p):
  return rearrange(x,"b(hw)(yx)c->b(hy)(wx)c",h=g,w=g,y=p,x=p)

x = MBConv(input) # MBConv layer

x = block(x,p) # window partition
x = RelSelfAttn(x) # Apply window-attention
x = unblock(x,x.shape[1]//p,p) # reverse

x = block(x,x.shape[1]//g) # grid partition
x = swapaxes(x,-2,-3) # move grid-axis to -2
x = RelSelfAttn(x) # Apply grid-attention
x = swapaxes(x,-2,-3) # reverse swapaxes
output = unblock(x,g,x.shape[1]//g) # reverse
```

fair comparison, we follow the original implementation without adopting any system-level strategies to further boost the final performance, such as the HTC framework [3], instaboost [8], *etc.* used in Swin [23]. We show the results of MaxViT-T/S/B on these two tasks to compare it against recent strong models at similar model complexity.

1.3 Image Aesthetics Model

This task requires incorporating both local and global information of an image to accurately predict human perceptual preference. To this end, the model needs to have the capacity to learn pixel-level quality aspects such as sharpness, noisiness and contrast as well as semantic-level aspects such as composition and depth-of-field. We follow [33] and use the normalized Earth Mover's Distance as our training loss. Given the ground truth and predicted probability mass functions $\mathbf{p}$ and $\widehat{\mathbf{p}}$ representing the histogram of scores, the normalized Earth Mover's Distance can be expressed as:

$$\mathrm{EMD}(\mathbf{p}, \widehat{\mathbf{p}}) = \left(\frac{1}{N} \sum_{k=1}^{N} |\mathrm{CDF}_{\mathbf{p}}(k) - \mathrm{CDF}_{\widehat{\mathbf{p}}}(k)|^r \right)^{1/r} \tag{8}$$

where $\mathrm{CDF}_{\mathbf{p}}(k)$ is the cumulative distribution function as $\sum_{i=1}^{k} \mathbf{p}_i$, and $N = 10$ represents the number score bins. In our experiments we set $r = 2$. We remove the classification head used in MaxViT, and instead append a fully-connected layer with 10 neurons followed by softmax.

Table 1: **Detailed architectural specifications** for MaxViT families.

	dsp. rate (out size)	MaxViT-T	MaxViT-S
stem	2× (112×112)	3×3, 64, stride 2 3×3, 64, stride 1	3×3, 64, stride 2 3×3, 64, stride 1
S1	4× (56 × 56)	[MBConv, 64, E 4, R 4 Rel-MSA, P 7×7, H 2 Rel-MSA, G 7×7, H 2] × 2	[MBConv, 96, E 4, R 4 Rel-MSA, P 7×7, H 3 Rel-MSA, G 7×7, H 3] × 2
S2	8× (28 × 28)	[MBConv, 128, E 4, R 4 Rel-MSA, P 7×7, H 4 Rel-MSA, G 7×7, H 4] × 2	[MBConv, 192, E 4, R 4 Rel-MSA, P 7×7, H 6 Rel-MSA, G 7×7, H 6] × 2
S3	16× (14 × 14)	[MBConv, 256, E 4, R 4 Rel-MSA, P 7×7, H 8 Rel-MSA, G 7×7, H 8] × 5	[MBConv, 384, E 4, R 4 Rel-MSA, P 7×7, H 12 Rel-MSA, G 7×7, H 12] × 5
S4	32× (7 × 7)	[MBConv, 512, E 4, R 4 Rel-MSA, P 7×7, H 16 Rel-MSA, G 7×7, H 16] × 2	[MBConv, 768, E 4, R 4 Rel-MSA, P 7×7, H 24 Rel-MSA, G 7×7, H 24] × 2
	dsp. rate (out size)	MaxViT-B	MaxViT-L
stem	2× (112×112)	3×3, 64, stride 2 3×3, 64, stride 1	3×3, 128, stride 2 3×3, 128, stride 1
S1	4× (56 × 56)	[MBConv, 96, E 4, R 4 Rel-MSA, P 7×7, H 3 Rel-MSA, G 7×7, H 3] × 2	[MBConv, 128, E 4, R 4 Rel-MSA, P 7×7, H 4 Rel-MSA, G 7×7, H 4] × 2
S2	8× (28 × 28)	[MBConv, 192, E 4, R 4 Rel-MSA, P 7×7, H 6 Rel-MSA, G 7×7, H 6] × 6	[MBConv, 256, E 4, R 4 Rel-MSA, P 7×7, H 8 Rel-MSA, G 7×7, H 8] × 6
S3	16× (14 × 14)	[MBConv, 384, E 4, R 4 Rel-MSA, P 7×7, H 12 Rel-MSA, G 7×7, H 12] ×14	[MBConv, 512, E 4, R 4 Rel-MSA, P 7×7, H 16 Rel-MSA, G 7×7, H 16] × 14
S4	32× (7 × 7)	[MBConv, 768, E 4, R 4 Rel-MSA, P 7×7, H 24 Rel-MSA, G 7×7, H 24] × 2	[MBConv, 1024, E 4, R 4 Rel-MSA, P 7×7, H 32 Rel-MSA, G 7×7, H 32] × 2

1.4 GAN Model

The above image recognition tasks can validate the power of our proposed MaxViT block used in downsampling (contracting) models. For this GAN experiment, we would like to demonstrate its effectiveness in upsampling (expanding) architectures. The MaxViT-GAN model for image generation is illustrated in Figure 2. For unconditional image generation, MaxViT-GAN first takes a latent code $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ as input, then progressively generates an image of target resolution through a hierarchically upsampling structure. We start by linearly projecting the input to a feature with spatial dimension 8×8. During the generation, the feature will go through five stages consisting of identical GAN blocks with gradually increased spatial resolution, similar to the design of our main model.

Similar to [44], we apply a cross-attention layer before the MaxViT block as a memory-efficient form of self-modulation in every stage, which has been shown to stabilize GAN training and also improve mode coverage [4, 44]. We use pixel shuffle [32] for upsampling in the end of each stage.

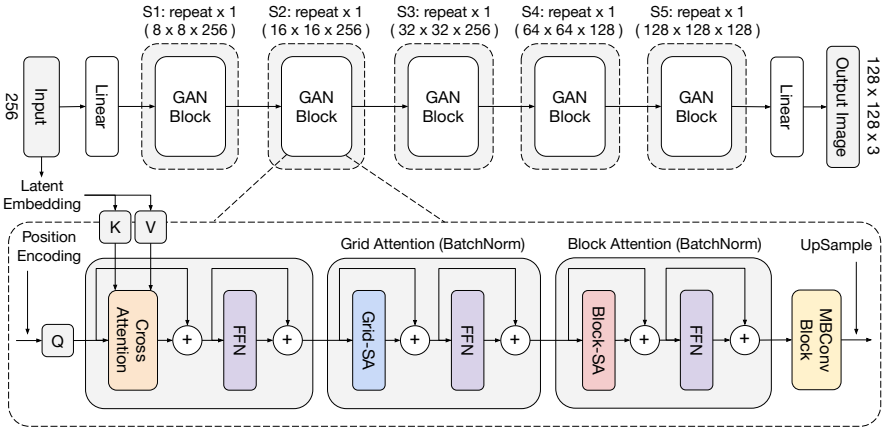

Fig. 2: **Generator architecture using the MaxViT block for the GAN experiment.** In every stage, we first use the cross-attention module to let the features attend to the latent embedding projected from the input code, which are then fed into the proposed MaxViT block consisting of grid attention, block attention, and MBConv layer. Note that unlike the main model in Sec. 1.1, the order of applying the three layers are reversed: from global to local.

2 Experimental Settings

2.1 ImageNet Classification

We provide ImageNet-1K experimental settings of MaxViT models for both pre-training and fine-tuning in Table 2. All the MaxViT variants used similar hyperparameters except that we mainly customize the stochastic depth rate to regularize each model separately.

2.2 Coco Detection and Segmentation

We evaluated MaxViT on the COCO2017 [22] object bounding box detection and instance segmentation tasks. The dataset contains 118K training and 5K validation samples. All the MaxViT backbones used are pretrained on ImageNet-1k at resolution 224×224. These pretrained checkpoints are then used as the warm-up weights for fine-tuning the detection and segmentation tasks. For both

Table 2: **Detailed hyperparameters used in ImageNet-1K experiments.** Multiple values separated by '/' are for each model size respectively.

Hyperparameter	**ImageNet-1K**		**ImageNet-21K**		**JFT-300M**	
	Pre-training (MaxViT-T/S/B/L)	Fine-tuning (MaxViT-T/S/B/L)	Pre-training (MaxViT-B/L/XL)	Fine-tuning (MaxViT-B/L/XL)	Pre-training (MaxViT-B/L/XL)	Fine-tuning (MaxViT-B/L/XL)
Stochastic depth	0.2/0.3/0.4/0.6		0.3/0.4/0.6	0.4/0.5/0.9	0.0/0.0/0.0	0.1/0.2/0.2
Center crop	True	False	True	False	True	False
RandAugment	2, 15	2, 15	2, 5	2, 15	2, 5	2, 15
Mixup alpha	0.8	0.8	None	None	None	None
Loss type	Softmax	Softmax	Sigmoid	Softmax	Sigmoid	Softmax
Label smoothing	0.1	0.1	0.0001	0.1	0	0.1
Train epochs	300	30	90	30	14	30
Train batch size	4096	512	4096	512	4096	512
Optimizer type	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Peak learning rate	3e-3	5e-5	1e-3	5e-5	1e-3	5e-5
Min learning rate	1e-5	5e-5	1e-5	5e-5	1e-5	5e-5
Warm-up	10K steps	None	5 epochs	None	20K steps	None
LR decay schedule	Cosine	None	Linear	None	Linear	None
Weight decay rate	0.05	1e-8	0.01	1e-8	0.01	1e-8
Gradient clip	1.0	1.0	1.0	1.0	1.0	1.0
EMA decay rate	None	0.9999	None	0.9999	None	0.9999

tasks, the input images are resized to 896×896. The training is conducted with a batch size of 256, using the AdamW [25] optimizer with learning rate of 1e-3, 3e-3, 3e-3, and stochastic depth of $0.8, 0.3, 0.3$ for MaxViT-T/S/B, respectively.

2.3 Image Aesthetics Assessment

We trained and evaluated the MaxViT model on the AVA benchmark [27]. This dataset consists of 255K images rated by armature photographers through photography contests. Each image is rated by an average of 200 human raters, assigning a score from 1 to 10 to images. The higher the score, the better the visual aesthetic quality of the image. Each image in the dataset has a histogram of scores associated with it, which we use as the ground truth label. Similar to [18, 33], we split the dataset into train and test sets, such that 20% of the data is used for testing. We train MaxViT for three different input resolutions: 224×224, 384×384 and 512×512. We initialized the model with ImageNet-1K 224×224 pre-trained weights. The weight and bias momentums are set to 0.9, and a dropout rate of 0.75 is applied on the last layer of the baseline network. We use an initial learning rate of 1e-3, exponentially decayed with decay factor 0.9 every 10 epochs. We set the stochastic depth rate to 0.5.

2.4 Image Generation

We use a ResNet-based discriminator following [17]. To train the model, we also used the standard non-saturating logistic GAN loss with $R1$ gradient penalty [26] applied to the discriminator with the gradient penalty weight set to 10. We employ the Adam [19] optimizer with a learning rate of 1e-4 for both generator and discriminator. The model is trained on TPU for one million steps with batch

size 256. Notably, we do not employ extra GAN training tricks such as pixel norm, noise injection, progressive growing, *etc.* on which recent state-of-the-art models are heavily relied to attain good results [16, 17]. The overall objectives of the GAN training are defined as:

$$\mathcal{L}_G = -\mathbb{E}_{z\sim P_z}[\log(D(G(z))], \tag{9}$$

$$\mathcal{L}_D = -\mathbb{E}_{x\sim P_x}[\log(D(x))] - \mathbb{E}_{z\sim P_z}[\log(1 - D(G(z)))] + \gamma\mathbb{E}_{x\sim P_x}[\|\nabla_x D(x)\|_2^2], \tag{10}$$

where γ denotes the R_1 gradient penalty weight.

3 Complete Experimental Results

We provide complete experiment comparisons for ImageNet-1K, Image-21K, and JFT datasets in Table 3 and Table 4, respectively. We also provide more visual results for unconditional image generation on ImageNet-1K in Figure 3.

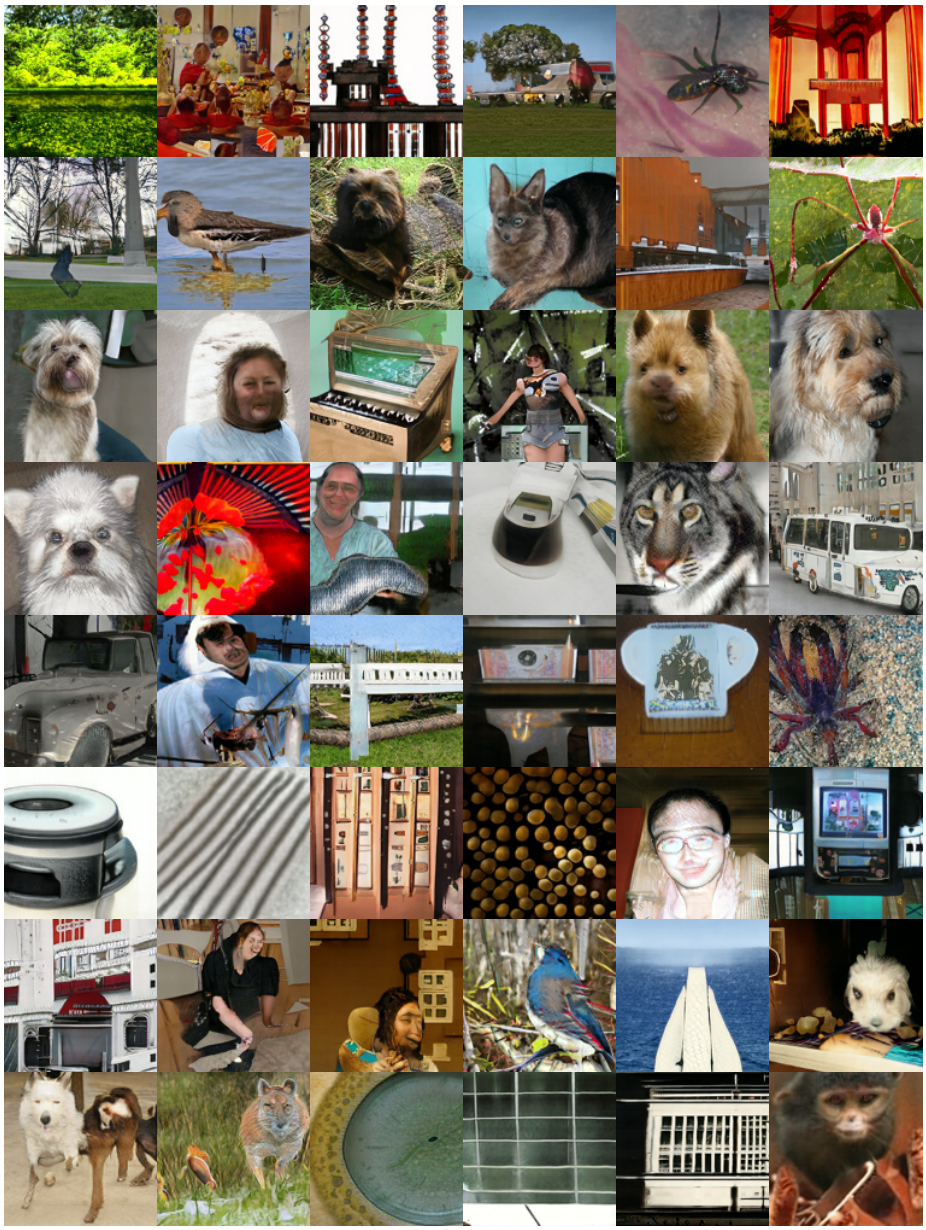

Fig. 3: Unconditional generation results on ImageNet-1k 128×128.

Table 3: Complete performance comparison under ImageNet-1K only setting.

	Model	Eval size	Params	FLOPs	throughput (img/s)	ImageNet top-1 acc.
ConvNets	•EffNet-B3 [34]	300	12M	1.8G	732.1	81.6
	•EffNet-B4 [34]	380	19M	4.2G	349.4	82.9
	•EffNet-B5 [34]	456	30M	9.9G	169.1	83.6
	•EffNet-B6 [34]	528	43M	19.0G	96.9	84.0
	•EffNet-B7 [34]	600	66M	37.0G	55.1	84.3
	•RegNetY-8GF [28]	224	39M	8.0G	591.6	81.7
	•RegNetY-16GF [28]	224	84M	16.0G	334.7	82.9
	•NFNet-F0 [2]	256	72M	12.4G	533,.3	83.6
	•NFNet-F1 [2]	320	132M	35.5G	228.5	84.7
	•NFNet-F2 [2]	352	194M	62.6G	129.0	85.1
	•NFNet-F3 [2]	416	255M	114.7G	78.8	85.7
	•NFNet-F4 [2]	512	316M	215.2G	51.7	85.9
	•NFNet-F5 [2]	544	377M	289.8G	-	86.0
	•EffNetV2-S [35]	384	24M	8.8G	666.6	83.9
	•EffNetV2-M [35]	380	55M	24.0G	280.7	85.1
	•EffNetV2-L [35]	480	121M	53.0G	163.2	85.7
	•ConvNeXt-T [24]	224	29M	4.5G	774.7	82.1
	•ConvNeXt-S [24]	224	50M	8.7G	447.1	83.1
	•ConvNeXt-B [24]	224	89M	15.4G	292.1	83.8
	•ConvNeXt-L [24]	384	198M	101.0G	50.4	85.5
ViTs	○ViT-B/32 [7]	384	86M	55.4G	85.9	77.9
	○ViT-B/16 [7]	384	307M	190.7G	27.3	76.5
	○DeiT-S [36]	224	22M	4.6G	940.4	79.8
	○DeiT-B [36]	224	86M	17.5G	292.3	81.8
	○DeiT-B [36]	384	86M	55.4G	85.9	83.1
	○CaiT-S36 [37]	224	68M	13.9G	-	83.3
	○CaiT-M24 [37]	224	186M	36.0G	-	83.4
	○CaiT-M24 [37]	384	186M	116.1G	-	84.5
	○DeepViT-S [45]	224	27M	6.2G	-	82.3
	○DeepViT-L [45]	224	55M	12.5G	-	83.1
	○T2T-ViT-14 [43]	224	22M	6.1G	-	81.7
	○T2T-ViT-19 [43]	224	39M	9.8G	-	82.2
	○T2T-ViT-24 [43]	224	64M	15.0G	-	82.6
	○Swin-T [23]	224	29M	4.5G	755.2	81.3
	○Swin-S [23]	224	50M	8.7G	436.9	83.0
	○Swin-B [23]	384	88M	47.0G	84.7	84.5
	○CSwin-B [6]	224	78M	15.0G	250	84.2
	○CSwin-B [6]	384	78M	47.0G	-	85.4
	○Focal-S [42]	224	51M	9.1G	-	83.5
	○Focal-B [42]	224	90M	16.0G	-	83.8
Hybrid	◇CvT-13 [41]	224	20M	4.5G	-	81.6
	◇CvT-21 [41]	224	32M	7.1G	-	82.5
	◇CvT-21 [41]	384	32M	24.9G	-	83.3
	◇CoAtNet-0 [5]	224	25M	4.2G	534.5	81.6
	◇CoAtNet-1 [5]	224	42M	8.4G	336.5	83.3
	◇CoAtNet-2 [5]	224	75M	15.7G	247.6	84.1
	◇CoAtNet-3 [5]	384	168M	107.4G	48.5	85.8
	◇CoAtNet-3 [5]	512	168M	203.1G	22.4	86.0
	◇MaxViT-T	224	31M	5.6G	349.6	83.62
	◇MaxViT-S	224	69M	11.7G	242.5	84.45
	◇MaxViT-B	224	120M	23.4G	133.6	84.95
	◇MaxViT-L	224	212M	43.9G	99.4	85.17
	◇MaxViT-T	384	31M	17.7G	121.9	83.62
	◇MaxViT-S	384	69M	36.1G	82.7	85.24
	◇MaxViT-B	384	120M	74.2G	45.8	85.74
	◇MaxViT-L	384	212M	133.1G	34.3	86.34
	◇MaxViT-T	512	31M	33.7G	63.8	85.72
	◇MaxViT-S	512	69M	67.6G	43.3	86.19
	◇MaxViT-B	512	120M	138.5G	24.0	86.66
	◇MaxViT-L	512	212M	245.4G	17.8	**86.70**

Table 4: Complete performance comparison for ImageNet-21K and JFT pre-trained models.

	Model	Eval size	Params	FLOPs	IN-1K top-1 acc.	
					21K→1K	JFT→1K
ConvNets	•BiT-R-101x3 [20]	384	388M	204.6G	84.4	
	•BiT-R-152x4 [20]	480	937M	840.5G	85.4	
	•EffNetV2-S [35]	384	24M	8.8G	85.0	
	•EffNetV2-M [35]	480	55M	24.0G	86.1	
	•EffNetV2-L [35]	480	121M	53.0G	86.8	
	•EffNetV2-XL [35]	512	208M	94.0G	87.3	
	•NFNet-F4+ [2]	512	527M	367G	-	89.20
	•ConvNeXt-B [24]	384	89M	45.1G	86.8	
	•ConvNeXt-L [24]	384	198M	101.0G	87.5	
	•ConvNeXt-XL [24]	384	350M	179.0G	87.8	
ViTs	○ViT-B/16 [7]	384	87M	55.5G	84.0	
	○ViT-L/16 [7]	384	305M	191.1G	85.2	
	○ViT-L/16 [7]	512	305M	364G	-	87.76
	○ViT-H/14 [7]	518	632M	1021G	-	88.55
	○HaloNet-H4 [38]	384	85M	-	85.6	
	○HaloNet-H4 [38]	512	85M	-	85.8	
	○Swin-B [23]	384	88M	47.0G	86.4	
	○Swin-L [23]	384	197M	103.9G	87.3	
	○SwinV2-B [23]	384	88M	-	87.1	
	○SwinV2-L [23]	384	197M	-	87.7	
	○CSwin-B [6]	384	78M	47.0G	87.0	
	○CSwin-L [6]	384	173M	96.8G	87.5	
Hybrid	◇CvT-13 [41]	384	20M	16.0G	83.3	
	◇CvT-21 [41]	384	32M	25.0G	84.9	
	◇CvT-W24 [41]	384	277M	193.2G	87.7	
	◇ResNet+ViT-L/16 [7]	384	330M	-	-	87.12
	◇CoAtNet-2 [5]	384	75M	49.8G	87.1	
	◇CoAtNet-3 [5]	384	168M	107.4G	87.6	
	◇CoAtNet-4 [5]	384	275M	189.5G	87.9	
	◇CoAtNet-2 [5]	512	75M	96.7G	87.3	
	◇CoAtNet-3 [5]	512	168M	203.1G	87.9	88.81
	◇CoAtNet-4 [5]	512	275M	360.9G	88.1	89.11
	◇CoAtNet-5 [5]	512	688M	812G	-	**89.77**
	◇MaxViT-B	384	119M	74.2G	88.24	88.69
	◇MaxViT-L	384	212M	128.7G	88.32	89.12
	◇MaxViT-XL	384	475M	293.7G	88.51	89.36
	◇MaxViT-B	512	119M	138.3G	88.38	88.82
	◇MaxViT-L	512	212M	245.2G	88.46	89.41
	◇MaxViT-XL	512	475M	535.2G	**88.70**	89.53

References

1. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016) 3
2. Brock, A., De, S., Smith, S.L., Simonyan, K.: High-performance large-scale image recognition without normalization. In: International Conference on Machine Learning. pp. 1059–1071. PMLR (2021) 10, 11
3. Chen, K., Pang, J., Wang, J., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Shi, J., Ouyang, W., Loy, C.C., Lin, D.: Hybrid task cascade for instance segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (2019) 4
4. Chen, T., Lucic, M., Houlsby, N., Gelly, S.: On self modulation for generative adversarial networks. arXiv preprint arXiv:1810.01365 (2018) 6
5. Dai, Z., Liu, H., Le, Q., Tan, M.: Coatnet: Marrying convolution and attention for all data sizes. Advances in Neural Information Processing Systems **34** (2021) 1, 2, 10, 11
6. Dong, X., Bao, J., Chen, D., Zhang, W., Yu, N., Yuan, L., Chen, D., Guo, B.: Cswin transformer: A general vision transformer backbone with cross-shaped windows. arXiv preprint arXiv:2107.00652 (2021) 10, 11
7. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020) 3, 10, 11
8. Fang, H.S., Sun, J., Wang, R., Gou, M., Li, Y.L., Lu, C.: Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 682–691 (2019) 4
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: 2017 IEEE International Conference on Computer Vision (ICCV). pp. 2980–2988 (2017). https://doi.org/10.1109/ICCV.2017.322 3
10. He, K., Zhang, X., Ren, S., Sun, J.: Identity mappings in deep residual networks. In: European conference on computer vision. pp. 630–645. Springer (2016) 1
11. Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415 (2016) 1
12. Ho, J., Kalchbrenner, N., Weissenborn, D., Salimans, T.: Axial attention in multidimensional transformers. arXiv preprint arXiv:1912.12180 (2019) 3
13. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7132–7141 (2018) 1
14. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. pp. 448–456. PMLR (2015) 1
15. Jiang, Y., Chang, S., Wang, Z.: Transgan: Two pure transformers can make one strong gan, and that can scale up. Advances in Neural Information Processing Systems **34** (2021) 1
16. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196 (2017) 8
17. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8110–8119 (2020) 7, 8

18. Ke, J., Wang, Q., Wang, Y., Milanfar, P., Yang, F.: Musiq: Multi-scale image quality transformer. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5148–5157 (2021) 7
19. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 7
20. Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., Houlsby, N.: Big transfer (bit): General visual representation learning. In: European conference on computer vision. pp. 491–507. Springer (2020) 11
21. Lin, T.Y., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 936–944 (2017) 3
22. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014) 6
23. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022 (2021) 1, 2, 3, 4, 10, 11
24. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. arXiv preprint arXiv:2201.03545 (2022) 10, 11
25. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017) 7
26. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for gans do actually converge? In: International conference on machine learning. pp. 3481–3490. PMLR (2018) 7
27. Murray, N., Marchesotti, L., Perronnin, F.: Ava: A large-scale database for aesthetic visual analysis. In: 2012 IEEE conference on computer vision and pattern recognition. pp. 2408–2415. IEEE (2012) 7
28. Radosavovic, I., Kosaraju, R.P., Girshick, R., He, K., Dollár, P.: Designing network design spaces. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10428–10436 (2020) 10
29. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 28. Curran Associates, Inc. (2015), https://proceedings.neurips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf 3
30. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018) 1
31. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. arXiv preprint arXiv:1803.02155 (2018) 1
32. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1874–1883 (2016) 6
33. Talebi, H., Milanfar, P.: Nima: Neural image assessment. IEEE transactions on image processing **27**(8), 3998–4011 (2018) 4, 7
34. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019) 1, 10

35. Tan, M., Le, Q.: Efficientnetv2: Smaller models and faster training. In: International Conference on Machine Learning. pp. 10096–10106. PMLR (2021) 10, 11
36. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers & distillation through attention. In: International Conference on Machine Learning. pp. 10347–10357. PMLR (2021) 10
37. Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., Jégou, H.: Going deeper with image transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 32–42 (2021) 10
38. Vaswani, A., Ramachandran, P., Srinivas, A., Parmar, N., Hechtman, B., Shlens, J.: Scaling local self-attention for parameter efficient visual backbones. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12894–12904 (2021) 1, 11
39. Wang, H., Zhu, Y., Green, B., Adam, H., Yuille, A., Chen, L.C.: Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In: European Conference on Computer Vision. pp. 108–126. Springer (2020) 3
40. Wu, F., Fan, A., Baevski, A., Dauphin, Y.N., Auli, M.: Pay less attention with lightweight and dynamic convolutions. arXiv preprint arXiv:1901.10430 (2019) 1
41. Wu, H., Xiao, B., Codella, N., Liu, M., Dai, X., Yuan, L., Zhang, L.: Cvt: Introducing convolutions to vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 22–31 (2021) 10, 11
42. Yang, J., Li, C., Zhang, P., Dai, X., Xiao, B., Yuan, L., Gao, J.: Focal self-attention for local-global interactions in vision transformers. arXiv preprint arXiv:2107.00641 (2021) 10
43. Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Jiang, Z.H., Tay, F.E., Feng, J., Yan, S.: Tokens-to-token vit: Training vision transformers from scratch on imagenet. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 558–567 (2021) 10
44. Zhao, L., Zhang, Z., Chen, T., Metaxas, D., Zhang, H.: Improved transformer for high-resolution gans. Advances in Neural Information Processing Systems **34** (2021) 6
45. Zhou, D., Kang, B., Jin, X., Yang, L., Lian, X., Jiang, Z., Hou, Q., Feng, J.: Deepvit: Towards deeper vision transformer. arXiv preprint arXiv:2103.11886 (2021) 10