EfficientNetV2：2021

Smaller Models and Fast Training

code:**https://github.com/google/automl/tree/master/efficientnetv2**

论文出发点或背景

随着模型规模和训练数据规模越来越大，训练效率对深度学习越来越重要。

本文的目标是比现有技术更显著地提高训练速度和参数效率

论文创新思路

对于不同的图像大小保持相同的正则化并不理想的：对于相同的网络，小的图像大小导致小的网络容量，因此需要弱正则化；反之，大的图像尺寸需要更强的正则化来对抗过拟合

不同类型的渐进式训练，可以动态地改变训练设置或网络。

与以往的工作不同，本文使用NAS来优化训练和参数效率。

论文方法介绍

为了开发这些模型，我们使用了训练感知的神经结构搜索和缩放的组合，以共同优化训练速度和参数效率。

我们的训练可以通过在训练过程中逐步增加图像的大小来进一步加快，但它经常会导致准确性的下降。为了弥补这种精度的下降，我们提出了一种改进的渐进学习方法，它可以自适应地调整正则化（例如，数据增强）和图像大小。

我们的研究在efficientnet中显示：(1)在非常大的图像大小的训练是慢的；(2)深度卷积在早期的层上比较缓慢。(3)同等地扩大每个阶段也是次优的。

我们提出了一种改进的渐进学习方法：在早期训练时代，我们用小图像大小和弱正则化（如退出和数据增强）训练网络，然后逐渐增加图像大小和更强的正则化。建立在逐步调整大小的基础上，但通过动态调整正则化，我们的方法可以在不导致精度下降的情况下加速训练。

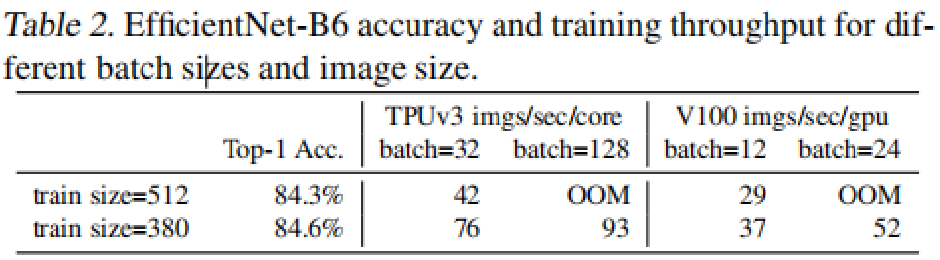


Table 2. EfficientNet-B6 accuracy and training throughput for different batch sizes and image size.

	Top-1 Acc.	TPUv3 imgs/sec/core		V100 imgs/sec/gpu	
		batch=32	batch=128	batch=12	batch=24
train size=512	84.3%	42	OOM	29	OOM
train size=380	84.6%	76	93	37	52

MBConv和fusens-MBConv的正确组合是很重要的，这激励我们利用神经结构搜索来自动搜索最佳组合。

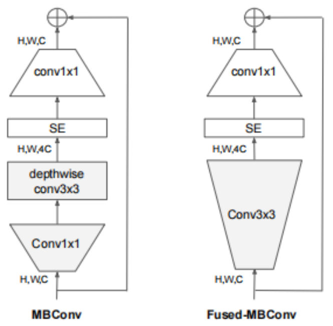


Figure 2. Structure of MBConv and Fused-MBConv.

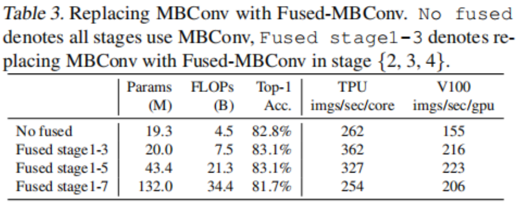


Table 3. Replacing MBConv with Fused-MBConv. `No fused` denotes all stages use MBConv, `Fused stage1-3` denotes replacing MBConv with Fused-MBConv in stage {2, 3, 4}.

	Params (M)	FLOPs (B)	Top-1 Acc.	TPU imgs/sec/core	V100 imgs/sec/gpu
No fused	19.3	4.5	82.8%	262	155
Fused stage1-3	20.0	7.5	83.1%	362	216
Fused stage1-5	43.4	21.3	83.1%	327	223
Fused stage1-7	132.0	34.4	81.7%	254	206

在本文中，我们将使用一种非均匀的缩放策略，逐步向后期阶段添加更多的层。

搜索到的EfficientNetv2，与EfficientNet相比。主要有如下区别：

(1):EfficientNetv2广泛地使用了MBConv并且在早期层新添加了fused-MBConv.

(2):EfficientNetv2更喜欢在MBConv上选择小的扩展比，导致了更少的内存占用。

(3):EfficientNetv2更喜欢3*3核，但是添加了更多的层用来补偿由较小的核而导致感受野的减少。

(4):EfficientNetv2完全删除了原始EfficientNet最后的stride-1 stage，或许是由于考虑到参数和内存占用问题。

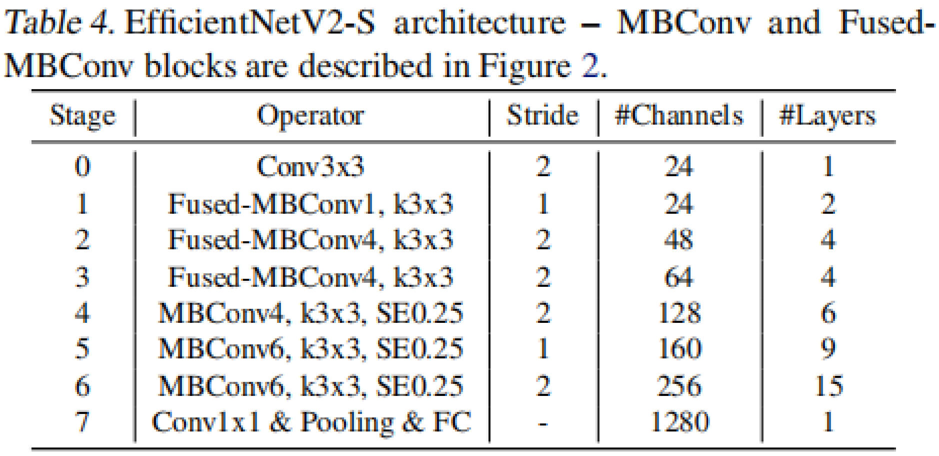
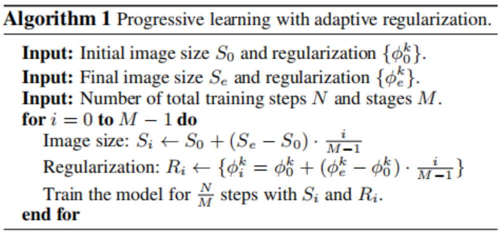


Table 4. EfficientNetV2-S architecture – MBConv and Fused-MBConv blocks are described in Figure 2.

Stage	Operator	Stride	#Channels	#Layers
0	Conv3x3	2	24	1
1	Fused-MBConv1, k3x3	1	24	2
2	Fused-MBConv4, k3x3	2	48	4
3	Fused-MBConv4, k3x3	2	64	4
4	MBConv4, k3x3, SE0.25	2	128	6
5	MBConv6, k3x3, SE0.25	1	160	9
6	MBConv6, k3x3, SE0.25	2	256	15
7	Conv1x1 & Pooling & FC	-	1280	1

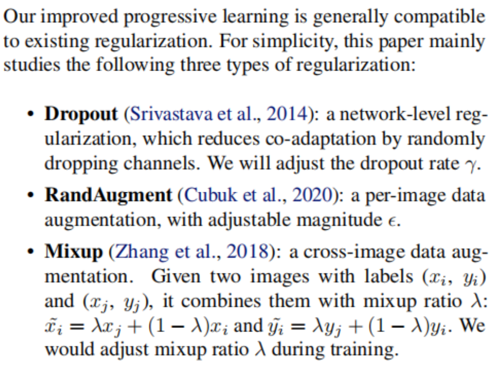
在本文中，我们认为，即使对于相同的网络，较小的图像尺寸也会导致较小的网络容量，因此需要较弱的正则化；反之，图像尺寸越大，计算量越大，容量越大，因此更容易过拟合。

在早期的训练阶段，我们用更小的图像和较弱的正则化来训练网络，这样网络就可以轻松、快速地学习简单的表示。然后，我们逐渐增加图像的大小，但也通过增加更强的正则化，使学习更加困难。



Algorithm 1 Progressive learning with adaptive regularization.

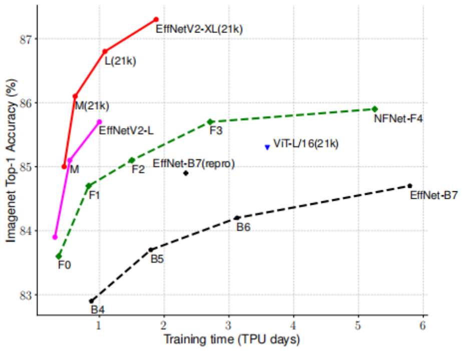
```
Input: Initial image size $S_0$ and regularization $\{\phi_0^k\}$.
Input: Final image size $S_e$ and regularization $\{\phi_e^k\}$.
Input: Number of total training steps $N$ and stages $M$.
for $i = 0$ to $M - 1$ do
    Image size: $S_i \leftarrow S_0 + (S_e - S_0) \cdot \frac{i}{M-1}$
    Regularization: $R_i \leftarrow \{\phi_i^k = \phi_0^k + (\phi_e^k - \phi_0^k) \cdot \frac{i}{M-1}\}$
    Train the model for $\frac{N}{M}$ steps with $S_i$ and $R_i$.
end for
```



Our improved progressive learning is generally compatible to existing regularization. For simplicity, this paper mainly studies the following three types of regularization:

- **Dropout** (Srivastava et al., 2014): a network-level regularization, which reduces co-adaptation by randomly dropping channels. We will adjust the dropout rate γ.
- **RandAugment** (Cubuk et al., 2020): a per-image data augmentation, with adjustable magnitude ϵ.
- **Mixup** (Zhang et al., 2018): a cross-image data augmentation. Given two images with labels (x_i, y_i) and (x_j, y_j), it combines them with mixup ratio λ: $\tilde{x}_i = \lambda x_j + (1 - \lambda)x_i$ and $\tilde{y}_i = \lambda y_j + (1 - \lambda)y_i$. We would adjust mixup ratio λ during training.

实际效果



(a) Training efficiency.

	EfficientNet (2019)	ResNet-RS (2021)	DeiT/ViT (2021)	EfficientNetV2 (ours)
Top-1 Acc.	84.3%	84.0%	83.1%	83.9%
Parameters	43M	164M	86M	24M

(b) Parameter efficiency.

Figure 1. **ImageNet ILSVRC2012 top-1 Accuracy vs. Training Time and Parameters –** Models tagged with `21k` are pretrained on ImageNet21k, and others are directly trained on ImageNet ILSVRC2012. Training time is measured with 32 TPU cores. All EfficientNetV2 models are trained with progressive learning. Our EfficientNetV2 trains 5x - 11x faster than others, while using up to 6.8x fewer parameters. Details are in Table 7 and Figure 5.

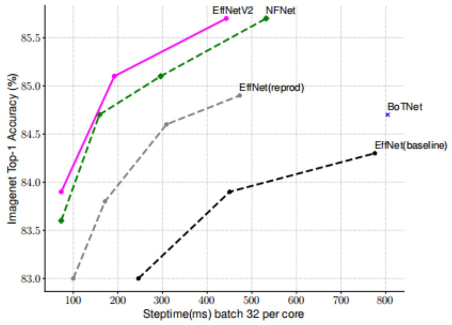


Figure 3. ImageNet accuracy and training step time on TPUv3 – Lower step time is better; all models are trained with fixed image size without progressive learning.

Table 6. Progressive training settings for EfficientNetV2.

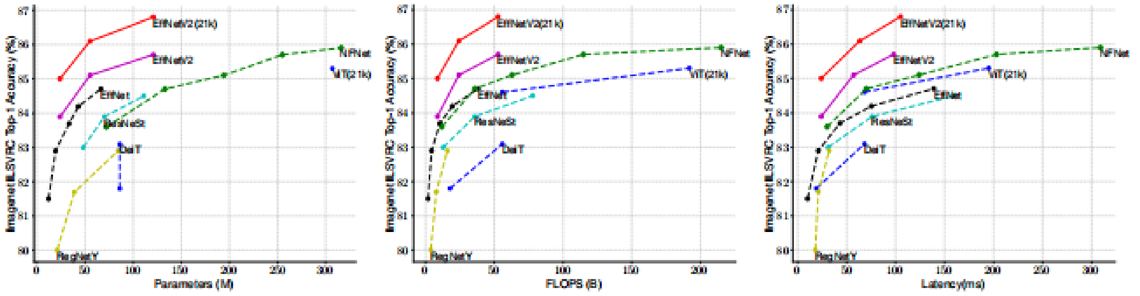
	S		M		L	
	min	max	min	max	min	max
Image Size	128	300	128	380	128	380
RandAugment	5	15	5	20	5	25
Mixup alpha	0	0	0	0.2	0	0.4
Dropout rate	0.1	0.3	0.1	0.4	0.1	0.5

Table 10. Comparison with the same training settings – Our new EfficientNetV2-M runs faster with less parameters.

	Acc. (%)	Params (M)	FLOPs (B)	TrainTime (h)	InferTime (ms)
V1-B7	85.0	66	38	54	170
V2-M (ours)	85.1	55 (-17%)	24 (-37%)	13 (-76%)	57 (-66%)

Table 8. **Transfer Learning Performance Comparison** – All models are pretrained on ImageNet ILSVRC2012 and finetuned on downstream datasets. Transfer learning accuracy is averaged over five runs.

	Model	Params	ImageNet Acc.	CIFAR-10	CIFAR-100	Flowers	Cars
ConvNets	GPipe (Huang et al., 2019)	556M	84.4	99.0	91.3	98.8	94.7
	EfficientNet-B7 (Tan & Le, 2019a)	66M	84.7	98.9	91.7	98.8	94.7
Vision Transformers	ViT-B/32 (Dosovitskiy et al., 2021)	88M	73.4	97.8	86.3	85.4	-
	ViT-B/16 (Dosovitskiy et al., 2021)	87M	74.9	98.1	87.1	89.5	-
	ViT-L/32 (Dosovitskiy et al., 2021)	306M	71.2	97.9	87.1	86.4	-
	ViT-L/16 (Dosovitskiy et al., 2021)	306M	76.5	97.9	86.4	89.7	-
	DeiT-B (ViT+regularization) (Touvron et al., 2021)	86M	81.8	99.1	90.8	98.4	92.1
	DeiT-B-384 (ViT+regularization) (Touvron et al., 2021)	86M	83.1	99.1	90.8	98.5	93.3
ConvNets (ours)	EfficientNetV2-S	24M	83.2	98.7±0.04	91.5±0.11	97.9±0.13	93.8±0.11
	EfficientNetV2-M	55M	85.1	99.0±0.08	92.2±0.08	98.5±0.08	94.6±0.10
	EfficientNetV2-L	121M	85.7	**99.1**±0.03	**92.3**±0.13	**98.8**±0.05	**95.1**±0.10



(a) Parameters (b) FLOPs (c) GPU V100 Latency (batch 16)

Figure 5. **Model Size, FLOPs, and Inference Latency** – Latency is measured with batch size 16 on V100 GPU. 21k denotes pretrained on ImageNet21k images, others are just trained on ImageNet ILSVRC2012. Our EfficientNetV2 has slightly better parameter efficiency with EfficientNet, but runs 3x faster for inference.

个人理解

两点创新：

1.得到了一个新的model——efficientnetV2，在结果上优于之前所有的网络

2.提出了一种改进的渐进学习方法：在早期训练时代，我们用小图像大小和弱正则化（如退出和数据增强）训练网络，然后逐渐增加图像大小和更强的正则化。建立在逐步调整大小的基础上，但通过动态调整正则化，我们的方法可以在不导致精度下降的情况下加速训练。

V2对比V1的话主要有以下改进

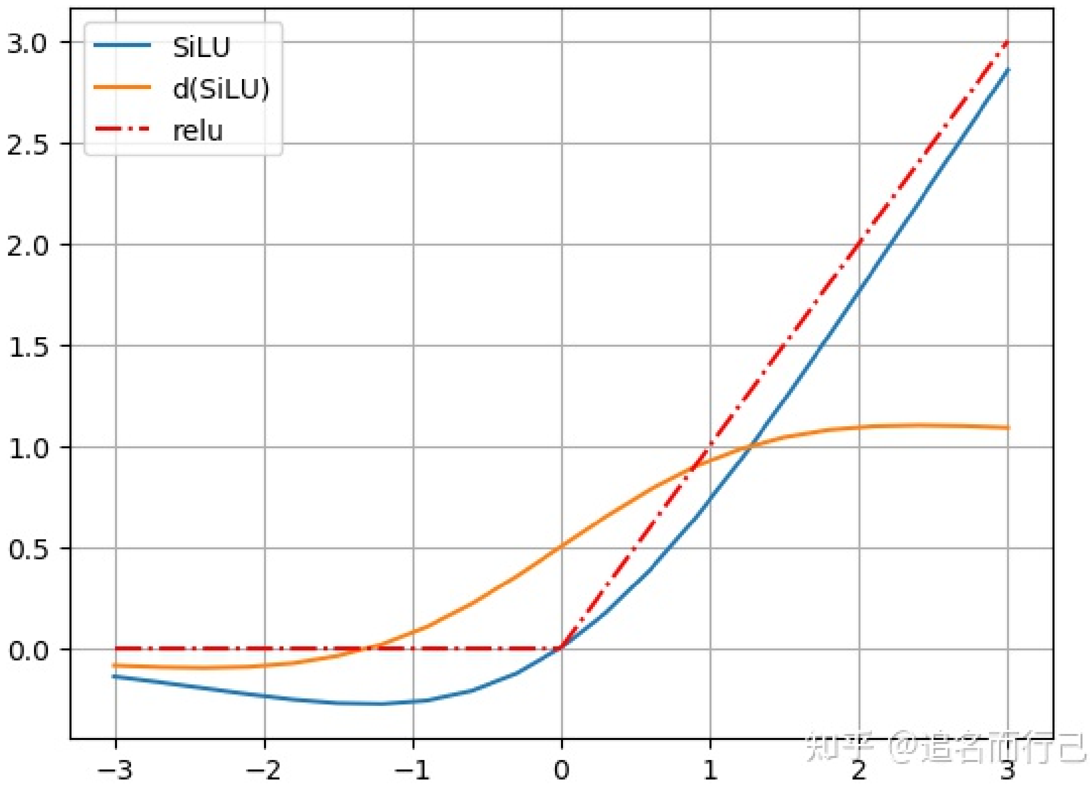
1.将网络前几层的MBConv改成了Fused-MBConv

2.使用较小的expansion ratio，可以减少内存访问开销

3.使用更小的kernel size（3×3）

4.移除了EfficientNetV1中最后一个步距为1的stage

补充以下该论文中的激活函数：silu (x)=x∗ sigmoid(x)


SiLU
d(SiLU)
relu
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−3
−2
−1
0
1
2
3