EfficientNet

摘要

卷积神经网络 (ConvNets) 通常在固定的资源预算下开发，如果有更多的资源可用，则按比例放大以获得更好的准确性。在本文中，我们系统地研究了模型缩放，并确定仔细平衡网络深度、宽度和分辨率可以带来更好的性能。基于这一观察，我们提出了一种新的缩放方法，该方法使用简单而有效的复合系数统一缩放深度/宽度/分辨率的所有维度。我们展示了这种方法在扩大 MobileNets 和 ResNet 方面的有效性。更进一步，我们使用神经架构搜索来设计一个新的基线网络并将其放大以获得一系列模型，称为 EfficientNets[1]，比以前的 ConvNets 实现了更好的准确性和效率。

关键词：卷积神经网络；残差网络；模型缩放

1 引言

在本文中，我们想研究和重新思考扩大 ConvNets 的过程。特别是，我们调查了中心问题：有一种原则性的方法可以扩大 ConvNets，以实现更好的准确性和效率。我们的实证研究表明，平衡网络宽度/深度/分辨率的所有维度是至关重要的，令人惊讶的是，这种平衡可以通过简单地用恒定的比率缩放它们中的每一个来实现。基于这一观察，我们提出了一种简单而有效的复合缩放方法。与传统的任意缩放这些因素的做法不同，我们的方法均匀地缩放网络宽度、深度、分辨率具有一组固定的缩放系数。例如，如果我们想使用 2N 次更多的计算资源，那么我们可以简单地将网络深度增加 N 、宽度 N 和图像大小增加 N ，其中 、 、 是常数系数，由对原始小模型进行小网格搜索确定。直观地说，复合缩放方法是有意义的，因为如果输入图像更大，那么网络需要更多的层来增加感受野和更多的通道来捕获更大图像上的更细粒度的模式。事实上，以前的理论都表明网络宽度和深度之间存在某种关系，但据我们所知，我们是第一个凭经验量化网络宽度、深度和分辨率所有三个维度之间的关系的人。

2 相关工作

此部分对课题内容相关的工作进行简要的分类概括与描述，二级标题中的内容为示意，可按照行文内容进行增删与更改，若二级标题无法对描述内容进行概括，可自行增加三级标题，后面内容同样如此，引文的 bib 文件统一粘贴到 **refs.bib** 中并采用如下引用方式 [1]。

2.1 模型缩放

有许多方法可以针对不同的资源约束缩放可以通过调整网络深度来缩小（例如 ResNet-18）或向上（例如 ResNet-200），而 WideResNet 可以通过网络宽度 (channels) 进行缩放。还很好地认识到，更大的输入图像大小将有助于准确性，但代价是更多的 FLOPS。尽管先前的研究表明网络深度和宽度对于 ConvNets 的表达能力都很重要，如何有效缩放 ConvNet 以获得更好的效率和准确性仍然是一个悬而未决的问题。我们的工作系统地和经验地研究了 ConvNet 对网络宽度、深度和分辨率的所有三个维度的缩放。

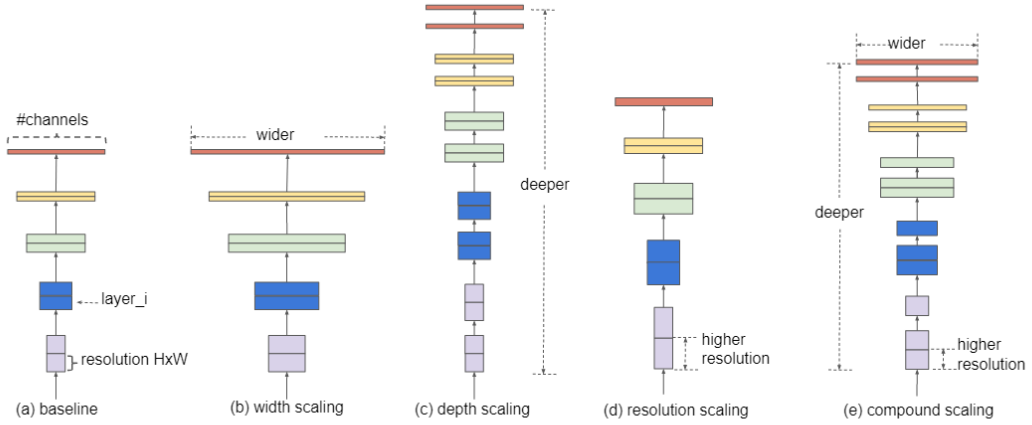

Figure 2. **Model Scaling.** (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

图 1. 分别在深度，宽度，分辨率三个维度的缩放图

2.2 卷积神经网络的效率

深度卷积网络经常被过度参数化。模型压缩是通过交易准确性以提高效率来减少模型大小的常用方法。随着手机的普及，手工制作高效的移动尺寸 ConvNets 也很常见，如 SqueezeNets，通过广泛调整网络宽度、深度、卷积核类型和大小，实现了比手工制作的移动 ConvNets 更好的效率。然而，尚不清楚如何将这些技术应用于具有更多设计空间和更昂贵调整成本的更大模型。在本文中，我们旨在研究超越最先进精度的超大型 ConvNets 的模型效率。为了实现这一目标，我们求助于模型缩放。

3 本文方法

3.1 本文方法概述

本文采用结合对深度，宽度，分辨率三个维度的组合扩大或缩放来进行提高网络训练效率和准确率

ConvNet 层 i 可以定义为函数：Yi =Fi(Xi)，其中 Fi 是算子，Yi 是输出张量，Xi 是输入张量，张量形状为 <Hi, Wi, Ci>1，其中 Hi 和 Wi 是空间维度，Ci 是通道维度。ConvNet N 可以由组合层列表表示：N = Fk... F2 F1(X1) = j=1...k Fj (X1)。在实践中，ConvNet 层通常划分为多个阶段，每个阶段的所有层共享相同的架构：例如，ResNet 有五个阶段，每个

阶段的所有层都具有相同的卷积类型，除了第一层执行下采样。因此，我们可以将 ConvNet 定义为：

$$\mathcal{N} = \bigoplus_{i=1...s} \mathcal{F}_i^{L_i}\left(X_{\langle H_i, W_i, C_i\rangle}\right)$$

图 2. 问题公式

与主要关注寻找最佳层架构 Fi 的常规 ConvNet 设计不同，模型缩放尝试在不改变基线网络中预定义的 Fi 的情况下扩展网络长度 (Li)、宽度 (Ci) 和/或分辨率 (Hi, Wi)。通过固定 Fi，模型缩放简化了新资源约束的设计问题，但它仍然是一个很大的设计空间来探索每一层的不同 Li、Ci、Hi、Wi。为了进一步减少设计空间，我们限制所有层必须以恒定比例均匀缩放。我们的目标是最大化任何给定资源约束的模型精度，这可以表述为优化问题：

$$\begin{aligned}
&\max_{d,w,r} \quad Accuracy\big(\mathcal{N}(d,w,r)\big) \\
&s.t. \quad \mathcal{N}(d,w,r) = \bigodot_{i=1...s} \hat{\mathcal{F}}_i^{d\cdot\hat{L}_i}\left(X_{\langle r\cdot\hat{H}_i, r\cdot\hat{W}_i, w\cdot\hat{C}_i\rangle}\right) \\
&\qquad \text{Memory}(\mathcal{N}) \le \text{target_memory} \\
&\qquad \text{FLOPS}(\mathcal{N}) \le \text{target_flops}
\end{aligned}$$

图 3. 优化公式

3.2 神经网络结构

下表为 EfficientNet-B0 的网络框架（B1-B7 就是在 B0 的基础上修改 Resolution，Channels 以及 Layers），可以看出网络总共分成了 9 个 Stage，第一个 Stage 就是一个卷积核大小为 3x3 步距为 2 的普通卷积层（包含 BN 和激活函数 Swish），Stage2 ~ Stage8 都是在重复堆叠 MBConv 结构（最后一列的 Layers 表示该 Stage 重复 MBConv 结构多少次），而 Stage9 由一个普通的 1x1 的卷积层（包含 BN 和激活函数 Swish）一个平均池化层和一个全连接层组成。表格中每个 MBConv 后会跟一个数字 1 或 6，这里的 1 或 6 就是倍率因子 n 即 MBConv 中第一个 1x1 的卷积层会将输入特征矩阵的 channels 扩充为 n 倍，其中 k3x3 或 k5x5 表示 MBConv 中 Depthwise Conv 所采用的卷积核大小。Channels 表示通过该 Stage 后输出特征矩阵的 Channels。

Table 1. **EfficientNet-B0 baseline network** – Each row describes a stage i with $\hat{L}_i$ layers, with input resolution $\langle \hat{H}_i, \hat{W}_i \rangle$ and output channels $\hat{C}_i$. Notations are adopted from equation 2.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3x3	224×224	32	1
2	MBConv1, k3x3	112×112	16	1
3	MBConv6, k3x3	112×112	24	2
4	MBConv6, k5x5	56×56	40	2
5	MBConv6, k3x3	28×28	80	3
6	MBConv6, k5x5	14×14	112	3
7	MBConv6, k5x5	14×14	192	4
8	MBConv6, k3x3	7×7	320	1
9	Conv1x1 & Pooling & FC	7×7	1280	1

图 4. 网格结构

MBConv 其实就是 MobileNetV3 网络中的 InvertedResidualBlock，但也有些许区别。一个是采用的激活函数不一样（EfficientNet 的 MBConv 中使用的都是 Swish 激活函数），另一个是在每个 MBConv 中都加入了 SE（Squeeze-and-Excitation）模块。下图是我自己绘制的 MBConv 结构。

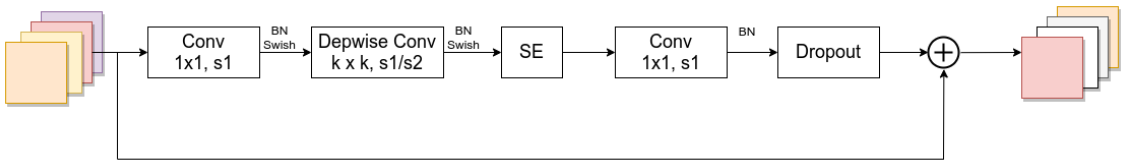

图 5. MBConv 结构

3.3 参数测试

在基准 EfficientNetB-0 上分别增加 width、depth 以及 resolution 后得到的统计结果。通过下图可以看出大概在 Accuracy 达到百分之八十就趋于饱和了

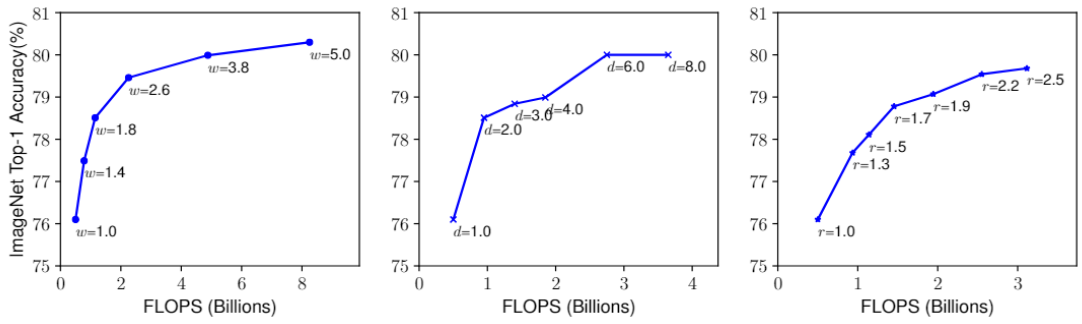

图 6. 单维度对比

采用不同的 d , r d, rd,r 组合，然后不断改变网络的 width 就得到了如下图所示的 4 条曲线，通过分析可以发现在相同的 FLOPs 下，同时增加 d 和 r 的效果最好。

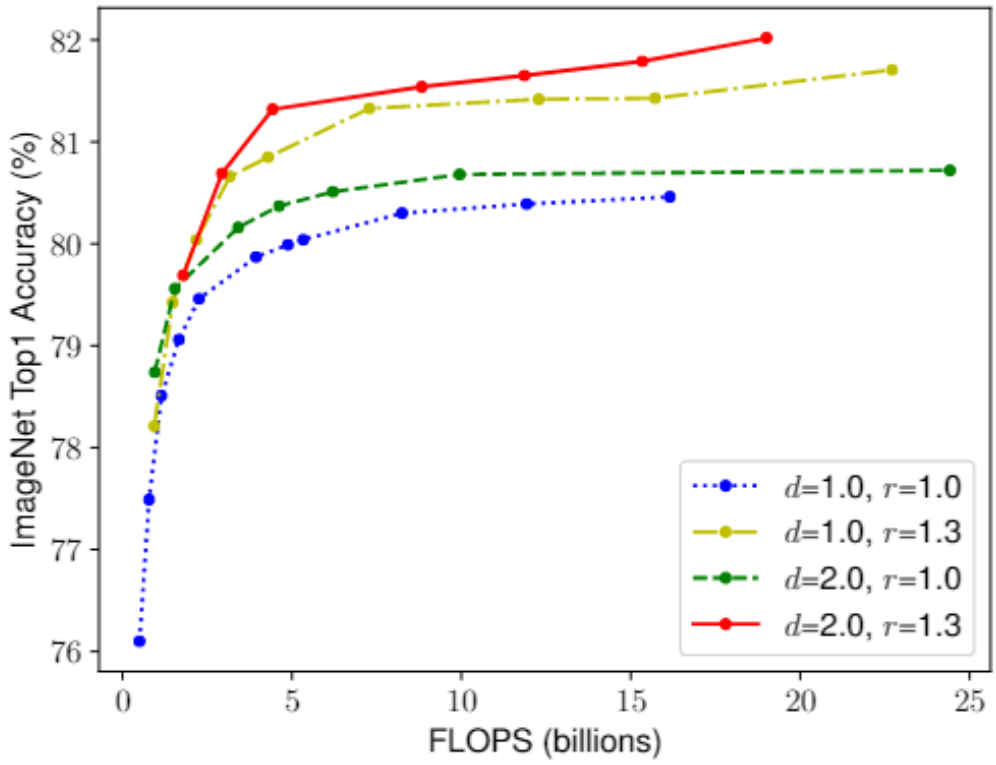

图 7. 维度组合对比

4 复现细节

4.1 与已有开源代码对比

根据网络结构的的特征，添加了改进加入恒等卷积和改变了激活函数，去保持图像特征，更好的优化网络结构，已有模型采用了 resnet 网络结构来加深训练层数，然后基于基本的卷积层和池化层组合，添加激活函数，来一步一步的构建 efficientNet 网络结构，实验参考了论文源码，基本的 efficientNet 的基本架构，在此基础上，进行改进.

4.2 实验环境搭建

编译器使用 PyCharm 安装环境所需 numpy matplotlib tqdm==4.56.0 torch>=1.7.1 torchvision>=0.8.2 python==3.9 数据集准备下载：[https://storage.googleapis.com/download.tensorflow.org/exa
$//storage.googleapis.com/download.tensorflow.org/example_images/flower_photos.tgz$)

4.3 创新点

添加了改进加入恒等卷积和改变了激活函数，去保持图像特征，更好的优化 efficientNet 网络结构. 添加恒等映射（Identity Mapping）和激活函数对网络的优势主要体现在以下几个方面：

网络深度的有效增加：在深度神经网络中，随着网络层数的增加，梯度消失和梯度爆炸等问题可能会出现，导致训练困难。通过添加恒等映射，可以提供一条直接的、无损的路径，使得梯度能够更容易地通过网络传播。这有助于有效增加网络的深度，使网络更容易训练和收敛。

信息流动的改善：添加恒等映射可以使得网络中的信息能够更自由地流动。在某些情况下，网络的某一层可能会丢失了一些有用的信息，而这些信息可能对后续层的学习很重要。通过引入恒等映射，可以保留和传递更多的信息，从而提高网络的表达能力和学习能力。

梯度传播的改善：在深度神经网络中，梯度的传播对于训练的成功非常重要。激活函数的选择对梯度的传播起着重要的作用。一些常用的激活函数如 ReLU 在负值区域存在梯度为零的问题，这可能导致信息丢失和梯度消失。通过选择合适的激活函数，如 Leaky ReLU、ELU 等，可以改善梯度的传播，避免梯度消失问题，从而提高网络的训练效果和性能。

4.4 论文实验分析

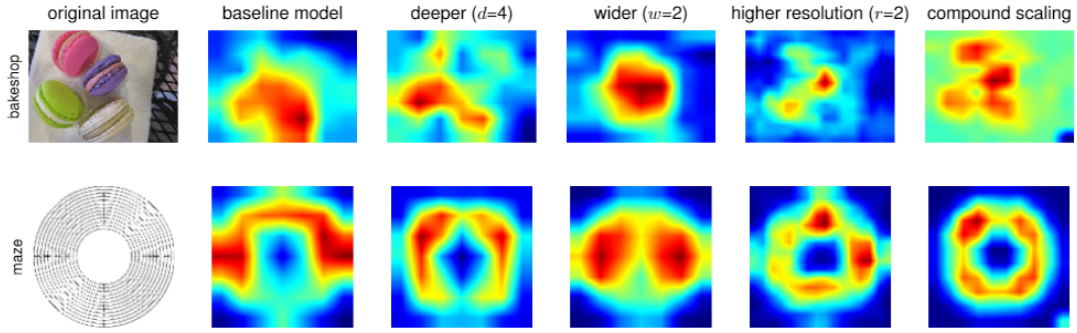

图 8. 具有不同缩放方法的模型的类激活图 (CAM)-我们的复合缩放方法允许缩放模型 (上一列) 关注具有更多对象细节的更相关区域。模型细节如表 7 所示。

	Comparison to best public-available results						Comparison to best reported results					
	Model	Acc.	#Param	Our Model	Acc.	#Param(ratio)	Model	Acc.	#Param	Our Model	Acc.	#Param(ratio)
CIFAR-10	NASNet-A	98.0%	85M	EfficientNet-B0	98.1%	4M (21x)	†Gpipe	**99.0%**	556M	EfficientNet-B7	98.9%	64M (8.7x)
CIFAR-100	NASNet-A	87.5%	85M	EfficientNet-B0	88.1%	4M (21x)	Gpipe	91.3%	556M	EfficientNet-B7	**91.7%**	64M (8.7x)
Birdsnap	Inception-v4	81.8%	41M	EfficientNet-B5	82.0%	28M (1.5x)	GPipe	83.6%	556M	EfficientNet-B7	**84.3%**	64M (8.7x)
Stanford Cars	Inception-v4	93.4%	41M	EfficientNet-B3	93.6%	10M (4.1x)	‡DAT	**94.8%**	-	EfficientNet-B7	94.7%	-
Flowers	Inception-v4	98.5%	41M	EfficientNet-B5	98.5%	28M (1.5x)	DAT	97.7%	-	EfficientNet-B7	**98.8%**	-
FGVC Aircraft	Inception-v4	90.9%	41M	EfficientNet-B3	90.7%	10M (4.1x)	DAT	92.9%	-	EfficientNet-B7	**92.9%**	-
Oxford-IIIT Pets	ResNet-152	94.5%	58M	EfficientNet-B4	94.8%	17M (5.6x)	GPipe	**95.9%**	556M	EfficientNet-B6	95.4%	41M (14x)
Food-101	Inception-v4	90.8%	41M	EfficientNet-B4	91.5%	17M (2.4x)	GPipe	93.0%	556M	EfficientNet-B7	**93.0%**	64M (8.7x)
Geo-Mean						**(4.7x)**						**(9.6x)**

图 9. 高效网在迁移学习数据集上的性能结果。我们的缩放 effentnet 模型在 8 个数据集中的 5 个数据集上实现了最新的精度，平均参数减少了 9.6 倍。

Model	Top-1 Acc.	Top-5 Acc.	#Params	Ratio-to-EfficientNet	#FLOPs	Ratio-to-EfficientNet
EfficientNet-B0	**77.1%**	**93.3%**	**5.3M**	**1x**	**0.39B**	**1x**
ResNet-50 (He et al., 2016)	76.0%	93.0%	26M	4.9x	4.1B	11x
DenseNet-169 (Huang et al., 2017)	76.2%	93.2%	14M	2.6x	3.5B	8.9x
EfficientNet-B1	**79.1%**	**94.4%**	**7.8M**	**1x**	**0.70B**	**1x**
ResNet-152 (He et al., 2016)	77.8%	93.8%	60M	7.6x	11B	16x
DenseNet-264 (Huang et al., 2017)	77.9%	93.9%	34M	4.3x	6.0B	8.6x
Inception-v3 (Szegedy et al., 2016)	78.8%	94.4%	24M	3.0x	5.7B	8.1x
Xception (Chollet, 2017)	79.0%	94.5%	23M	3.0x	8.4B	12x
EfficientNet-B2	**80.1%**	**94.9%**	**9.2M**	**1x**	**1.0B**	**1x**
Inception-v4 (Szegedy et al., 2017)	80.0%	95.0%	48M	5.2x	13B	13x
Inception-resnet-v2 (Szegedy et al., 2017)	80.1%	95.1%	56M	6.1x	13B	13x
EfficientNet-B3	**81.6%**	**95.7%**	**12M**	**1x**	**1.8B**	**1x**
ResNeXt-101 (Xie et al., 2017)	80.9%	95.6%	84M	7.0x	32B	18x
PolyNet (Zhang et al., 2017)	81.3%	95.8%	92M	7.7x	35B	19x
EfficientNet-B4	**82.9%**	**96.4%**	**19M**	**1x**	**4.2B**	**1x**
SENet (Hu et al., 2018)	82.7%	96.2%	146M	7.7x	42B	10x
NASNet-A (Zoph et al., 2018)	82.7%	96.2%	89M	4.7x	24B	5.7x
AmoebaNet-A (Real et al., 2019)	82.8%	96.1%	87M	4.6x	23B	5.5x
PNASNet (Liu et al., 2018)	82.9%	96.2%	86M	4.5x	23B	6.0x
EfficientNet-B5	**83.6%**	**96.7%**	**30M**	**1x**	**9.9B**	**1x**
AmoebaNet-C (Cubuk et al., 2019)	83.5%	96.5%	155M	5.2x	41B	4.1x
EfficientNet-B6	**84.0%**	**96.8%**	**43M**	**1x**	**19B**	**1x**
EfficientNet-B7	**84.3%**	**97.0%**	**66M**	**1x**	**37B**	**1x**
GPipe (Huang et al., 2018)	84.3%	97.0%	557M	8.4x	-	-

图 10. ImageNet 上的 EfficientNet 性能结果。所有的 effentnet 模型都是从我们的基线 effentnet - b0 开始，使用公式 3 中不同的复合系数 进行缩放。具有相似 top-1/top-5 精度的卷积神经网络被分组在一起进行效率比较。与现有的 ConvNets 相比，我们的规模化 effentnet 模型始终如一地将参数和 FLOPS 降低了一个数量级 (最多减少 8.4 倍参数，最多减少 16 倍 FLOPS)。

5 实验结果分析

根据所构建的 efficientNet 网络做的分类任务，可以看到，推理的结果非常好，在本文中，我们系统地研究了 ConvNet 缩放，并确定仔细平衡网络宽度、深度和分辨率是一个重要但缺失的部分，阻碍了我们更好的准确性和效率。为了解决这个问题，我们提出了一种简单有效的复合缩放方法，使我们能够以更有原则的方式轻松地将基线 ConvNet 扩展到任何目标资源约束，同时保持模型效率。受这种复合缩放方法的启发，我们证明了在 ImageNet 和五个常用的迁移学习数据集上，移动大小的 EfficientNet 模型可以非常有效地扩展，以更少的参数和 FLOPS 数量级超过了最先进的准确性。在本文中，我们系统地研究了卷积神经网络的缩放，并确定仔细平衡网络宽度，深度和分辨率是一个重要但缺失的部分，阻碍了我们更好的准确性和效率。为了解决这个问题，我们提出了一种简单而高效的复合缩放方法，该方法使我们能够以更有原则的方式轻松地将基线 ConvNet 扩展到任何目标资源约束，同时保持模型效率。在这种复合缩放方法的支持下，我们证明了在 ImageNet 和五种常用的迁移学习数据集上，mobilesize effentnet 模型可以非常有效地缩放，以更少的参数和 FLOPS 超过最先进的精度。

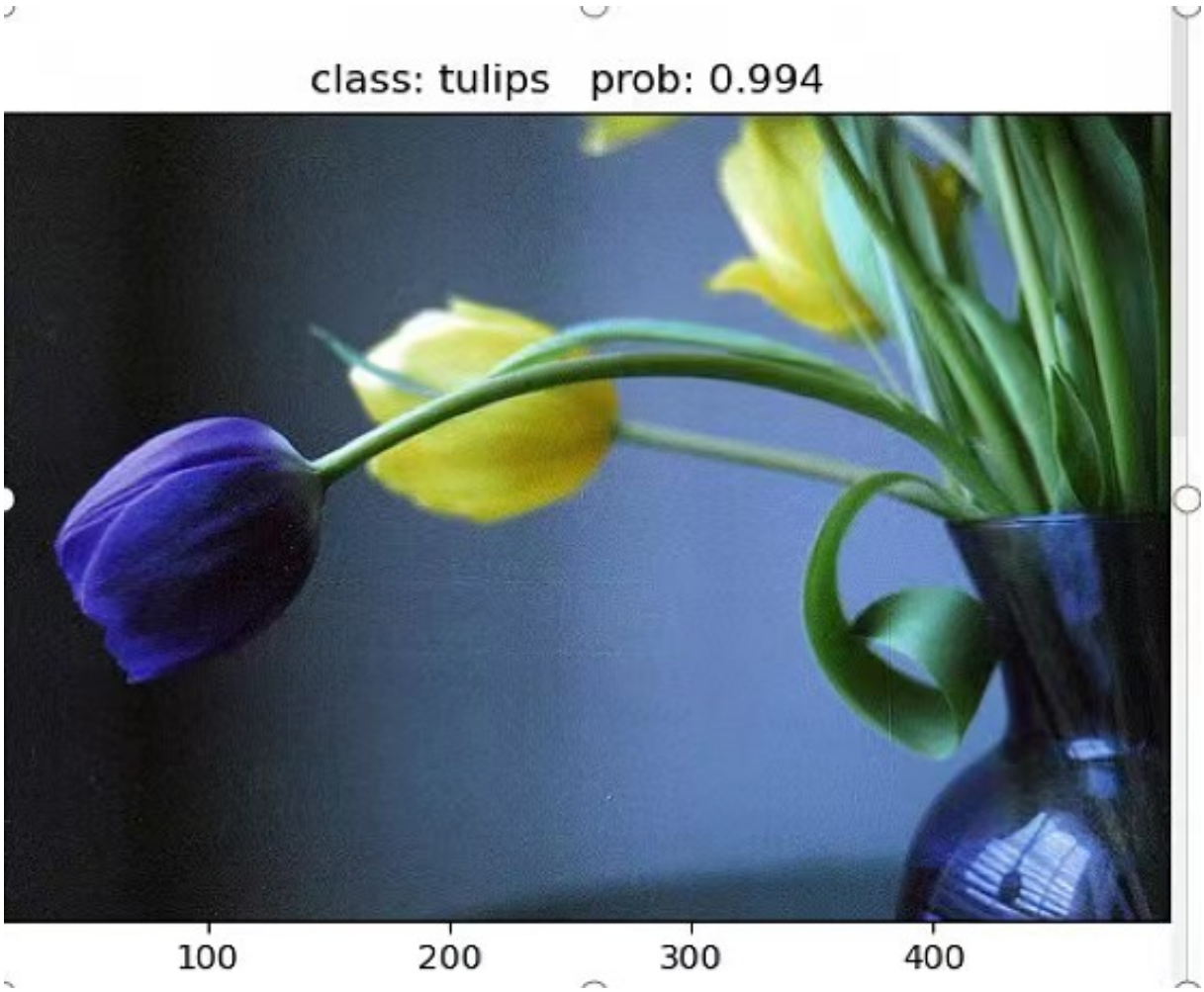

图 11. 图片分类结果

6 总结与展望

EfficientNet 是一种高效且准确的卷积神经网络架构，它通过改进网络的深度、宽度和分辨率来实现更好的性能和效率平衡。以下是对 EfficientNet 的总结和展望：总结：深度、宽度和分辨率的扩展：EfficientNet 通过统一缩放网络的深度、宽度和分辨率，以平衡模型的性能和计算资源的消耗。通过使用复合系数来扩展这三个维度，EfficientNet 在各种计算资源约束下都能获得较好的性能表现。引入了 MBConv 块：EfficientNet 采用了一种称为 MBConv（Mobile Inverted Residual Bottleneck）的特殊块结构，它结合了轻量级的深度可分离卷积和残差连接，以提高模型的表达能力和学习能力。自动网络缩放：EfficientNet 提出了一种自动网络缩放方法，通过在不同深度、宽度和分辨率的模型中进行均衡搜索，找到最优的网络结构。这种方法可以根据计算资源的不同，自动调整网络的规模，实现最佳的性能和效率平衡。展望：针对特定任务的改进：尽管 EfficientNet 在各种计算资源限制下表现出色，但仍有改进的空间。未来的研究可以探索针对特定任务的改进，例如在面部识别、目标检测和语义分割等领域进一步优化 EfficientNet 的性能。模型的轻量化：虽然 EfficientNet 已经是一种高效的网络架构，但在某些资源受限的环境下，仍需要更轻量的模型。因此，对 EfficientNet 进行更深入的轻量化研究，以提高其在移动设备和嵌入式系统上的实际可用性，是一个有前景的方向。跨模态学习：EfficientNet 主要应用于图像识别任务，但随着多模态数据（如图像、文本、语

音）的广泛应用，将 EfficientNet 扩展到跨模态学习是一个有潜力的研究方向。通过融合多种数据源和模态，可以进一步提高 EfficientNet 在多模态场景下的性能。总体而言，EfficientNet 是一种有效的网络架构，平衡了模型的性能和计算资源的消耗。未来的研究可以进一步探索 EfficientNet 在特定任务、轻量化以及跨模态学习等方面的改进，以满足不断发展的应用需求

参考文献

[1] Mingxing Tan Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. *ICML*, 32(11):1231–1237, 2019.