VGG 阅读总结

15331416 赵寒旭

目录

引言

VGGNet 是牛津大学计算机视觉组(Visual Geometry Group)和 Google Deepmind 公司研究员一起研发的深度卷积神经网络。VGGNet 在 AlexNet 的基础上探索了卷积神经网络的深度与性能之间的关系，通过反复堆叠 3*3 的小型卷积核和 2*2 的最大池化层，VGGNet 构筑的 16~19 层卷积神经网络模型取得了很好的识别性能，同时 VGGNet 的拓展性很强，迁移到其他图片数据上泛化能力很好，而且 VGGNet 结构简洁，现在依然被用来提取图像特征。

/*
注：部分理解内容来源于网上的 blog，
*/

VGG 的特点

1. 针对网络架构

1)全部使用3×3的卷积核和2×2的池化核，通过不断加深网络结构来提升性能。
2）使用多个小卷积核串联组成卷积层，和先前的大卷积核相比，拥有同样的感受野，却有着更少的参数，更强的非线性变换，因此有着更强的特征提取能力。
3）使用1×1卷积层，意义在于线性变换，输入通道和输出通道数不变，没有发生降维。
4）LRN（局部响应归一化）层作用不大（并不能提升性能，且会增加计算量），仅在一个网络中使用。

2. 针对过拟合现象

1）数据增强

使用 Multi-scale 的方法，将原始图像缩放到不同尺寸，再随机裁取固定大小的图片，这样可以增加很多数据量，防止模型过拟合。

2）采用 multi-crop 和 dense evaluation 配合（两者配合使用，效果比单使用好），可以很好的提升模型的性能。

3. 针对训练速度

可以先训练低深度的 A 网络，再复用 A 网络的权重初始化后面的几个复杂模型，这样训练收敛的速度更快

论文详细阅读

Abstract

研究了卷积网络深度对大型图像识别准确性的影响，主要评估小的卷积核（3×3）同等架构下随着网络深度的增加卷积网络的性能变化，随着网络深度达到 16-19 层，网络的性能也有着显著的提升。

主要的贡献：展示出网络的深度是算法优良性能的关键部分(使用了非常小的滤波器（3×3）。

1. Introduction

With ConvNets becoming more of a commodity in the computer vision field, a number of attempts have been made to improve the original architecture of Krizhevsky et al. (2012) in a bid to achieve better accuracy. For instance, the best-performing submissions to the ILSVRC-2013 (Zeiler & Fergus, 2013; Sermanet et al., 2014) utilised smaller receptive window size and smaller stride of the first convolutional layer. Another line of improvements dealt with training and testing the networks densely over the whole image and over multiple scales (Sermanet et al., 2014; Howard, 2014). In this paper, we address another important aspect of ConvNet architecture design – its depth. To this end, we fix other parameters of the architecture, and steadily increase the depth of the network by adding more convolutional layers, which is feasible due to the use of very small (3×3) convolution filters in all layers.

As a result, we come up with significantly more accurate ConvNet architectures, which not only achieve the state-of-the-art accuracy on ILSVRC classification and localisation tasks, but are also applicable to other image recognition datasets, where they achieve excellent performance even when used as a part of a relatively simple pipelines (e.g. deep features classified by a linear SVM without fine-tuning). We have released our two best-performing models[1] to facilitate further research.

为在 AlexNet 原始架构上实现更高的准确率，可以在第一个卷积层上使用更小的感受野和更短的步长，另一种路线是使用多尺度的密集训练和测试网络（后文讨论了 Multi-crop 和 Multi-scale 的使用对模型性能的影响）。
本文主要研究卷积神经网络架构设计的另一个重要影响因素：深度。
结果：实现了在 ILSVRC 上更精确的卷积神经网络，在其他数据集上也有很好的表现，迁移性很强。

2. ConvNet Configurations（卷积网络配置）

为在公平的环境下测量网络深度增加带来的提升，我们所有的卷积网络层结构使用相同的原则来设计。
本节介绍网络配置的细节。

2.1 Architecture

During training, the input to our ConvNets is a fixed-size 224×224 RGB image. The only pre-processing we do is subtracting the mean RGB value, computed on the training set, from each pixel. The image is passed through a stack of convolutional (conv.) layers, where we use filters with a very small receptive field: 3×3 (which is the smallest size to capture the notion of left/right, up/down, center). In one of the configurations we also utilise 1×1 convolution filters, which can be seen as a linear transformation of the input channels (followed by non-linearity). The convolution stride is fixed to 1 pixel; the spatial padding of conv. layer input is such that the spatial resolution is preserved after convolution, i.e. the padding is 1 pixel for 3×3 conv. layers. Spatial pooling is carried out by five max-pooling layers, which follow some of the conv. layers (not all the conv. layers are followed by max-pooling). Max-pooling is performed over a 2×2 pixel window, with stride 2.

A stack of convolutional layers (which has a different depth in different architectures) is followed by three Fully-Connected (FC) layers: the first two have 4096 channels each, the third performs 1000-way ILSVRC classification and thus contains 1000 channels (one for each class). The final layer is the soft-max layer. The configuration of the fully connected layers is the same in all networks.

All hidden layers are equipped with the rectification (ReLU (Krizhevsky et al., 2012)) non-linearity. We note that none of our networks (except for one) contain Local Response Normalisation (LRN) normalisation (Krizhevsky et al., 2012): as will be shown in Sect. 4, such normalisation does not improve the performance on the ILSVRC dataset, but leads to increased memory consumption and computation time. Where applicable, the parameters for the LRN layer are those of (Krizhevsky et al., 2012).

1）input：a fixed-size 224×224 RGB image
pre-processing：从每个像素减去 RGB 值的均值
2）The image is passed through a stack of convolutional layers:

使用非常小的3×3的滤波器（是足够捕获一个区域信息的最小的卷积核）
其中一个配置中使用1×1的卷积滤波器（对输入通道的线性变换），这样的卷积不改变输入通道的维度，且可以提高模型的学习能力。
convolution stride：1 pixel
spatial padding：对卷积层输入的扩展要保证卷积操作前后分辨率一致（eg. 3×3的卷积层要扩展 1 pixel）
Spatial pooling：由 5 个最大池化层实现（跟在某些卷积层后面）
Max-pooling：在2×2的窗口进行，步长为 2

3）3 Full-Connect layers
4096,4096,1000
最后一层是 softmax 层。

5）Hidden layers
整流函数：ReLU

6）Local Response Normalisation
除了一个网络之外，我们所有的网络都不含局部响应归一化，它不能提升在 ILSVRC 数据集上的性能，但会导致内存消耗和计算时间的增加。

7）LRN layer parameters
Where applicable, the parameters for the LRN layer are those of (Krizhevsky et al., 2012).

2.2 Configurations

卷积核尺寸很小，卷积核的数目初始为 64，每过一次 max-pooling 层，卷积核数目翻倍，最终增加到 512 个。
尽管网络深度很大，网络中的权重数目并不会比有更大卷积层和感受野的更浅的网络多。

2.3 Discussion

2013; Sermanet et al., 2014)), we use very small 3×3 receptive fields throughout the whole net, which are convolved with the input at every pixel (with stride 1). It is easy to see that a stack of two

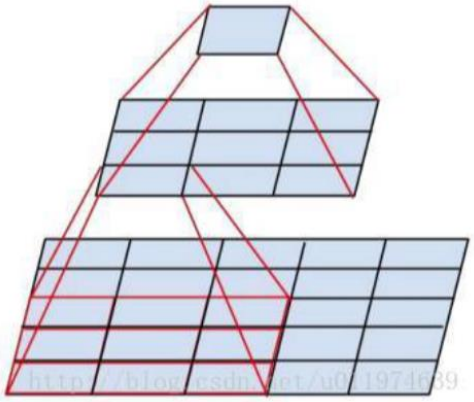
---

such layers have a 7×7 effective receptive field. So what have we gained by using, for instance, a stack of three 3×3 conv. layers instead of a single 7×7 layer? First, we incorporate three non-linear rectification layers instead of a single one, which makes the decision function more discriminative. Second, we decrease the number of parameters: assuming that both the input and the output of a three-layer 3×3 convolution stack has C channels, the stack is parametrised by $3\left(3^2C^2\right) = 27C^2$ weights; at the same time, a single 7×7 conv. layer would require $7^2C^2 = 49C^2$ parameters, i.e. 81% more. This can be seen as imposing a regularisation on the 7×7 conv. filters, forcing them to have a decomposition through the 3×3 filters (with non-linearity injected in between).

The incorporation of 1×1 conv. layers (configuration C, Table 1) is a way to increase the non-linearity of the decision function without affecting the receptive fields of the conv. layers. Even though in our case the 1×1 convolution is essentially a linear projection onto the space of the same dimensionality (the number of input and output channels is the same), an additional non-linearity is introduced by the rectification function. It should be noted that 1×1 conv. layers have recently been utilised in the "Network in Network" architecture of Lin et al. (2014).

使用多个小卷积核堆叠，感受野与大卷积核相同。
相比于单层，使用三个非线性整流层会使决策函数更有判别力。（多个小卷积核的堆叠会比单个大卷积核有更多激活函数的转换（ReLU），学习能力更强）
同时，多个小卷积核会有更少的参数。

使用1 × 1的卷积核，可以在不影响卷积层感受野的情况下增强决策函数的非线性，即使在本文模型使用1 × 1卷积核是同维度上的线性变换。（1 × 1卷积后会经过 ReLU 处理，而 ReLU 在输入大于 0 是线性的）



左图是两个 3*3 的卷积核堆叠在一起，可以看到:

在提取特征的能力上: 最顶层的 1*1 输出是由最底层的 5*5 的输入提取出来的，经过两个 3*3 的卷积核，效果与一个 5*5 的大卷积核类似。

在参数量上:两个 3*3 的卷积核，有 2*3*3=18 参数，而一个 5*5=25 参数，可以看到参数量上也少了很多

Table 1: **ConvNet configurations** (shown in columns). The depth of the configurations increases from the left (A) to the right (E), as more layers are added (the added layers are shown in bold). The convolutional layer parameters are denoted as "conv⟨receptive field size⟩-⟨number of channels⟩". The ReLU activation function is not shown for brevity.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

3. Classification Framework（分类框架）

本节介绍卷积神经网络训练和评估的细节。

3.1 Training

The ConvNet training procedure generally follows Krizhevsky et al. (2012) (except for sampling the input crops from multi-scale training images, as explained later). Namely, the training is carried out by optimising the multinomial logistic regression objective using mini-batch gradient descent (based on back-propagation (LeCun et al., 1989)) with momentum. The batch size was set to 256, momentum to 0.9. The training was regularised by weight decay (the L_2 penalty multiplier set to $5 \cdot 10^{-4}$) and dropout regularisation for the first two fully-connected layers (dropout ratio set to 0.5). The learning rate was initially set to 10^{-2}, and then decreased by a factor of 10 when the validation set accuracy stopped improving. In total, the learning rate was decreased 3 times, and the learning was stopped after 370K iterations (74 epochs). We conjecture that in spite of the larger number of parameters and the greater depth of our nets compared to (Krizhevsky et al., 2012), the nets required less epochs to converge due to (a) implicit regularisation imposed by greater depth and smaller conv. filter sizes; (b) pre-initialisation of certain layers.

训练过程基本按照 AlexNet 的步骤进行（除了 sampling the input crops from multi-scale training images）使用基于反向传播的小批量梯度下降来优化多项逻辑回归目标进行训练。
尽管与 AlexNet 相比，网络的参数数量更多，网络深度更大，但网络达到收敛需要的迭代次数更少，原因在于：
(a) 由更大的网络深度和更小的卷积层滤波器尺寸带来的隐式正则化。
(b) 某些层的预初始化

The initialisation of the network weights is important, since bad initialisation can stall learning due to the instability of gradient in deep nets. To circumvent this problem, we began with training the configuration A (Table 1), shallow enough to be trained with random initialisation. Then, when training deeper architectures, we initialised the first four convolutional layers and the last three fully-connected layers with the layers of net A (the intermediate layers were initialised randomly). We did not decrease the learning rate for the pre-initialised layers, allowing them to change during learning. For random initialisation (where applicable), we sampled the weights from a normal distribution with the zero mean and 10^{-2} variance. The biases were initialised with zero. It is worth noting that after the paper submission we found that it is possible to initialise the weights without pre-training by using the random initialisation procedure of Glorot & Bengio (2010).

To obtain the fixed-size 224×224 ConvNet input images, they were randomly cropped from rescaled training images (one crop per image per SGD iteration). To further augment the training set, the crops underwent random horizontal flipping and random RGB colour shift (Krizhevsky et al., 2012). Training image rescaling is explained below.

网络权重的初始化很重要，因为深层网络中的梯度不稳定，初始化不好可能会导致学习停滞。
为此先训练一个深度较浅的网络 A，再利用已经训练好的 A 网络权值初始化深度较深的网络。（迁移学习）
初始化权重从均值为 0 方差为 0.01 的正态分布中采样。
为了获得固定大小的224 × 224的卷积网络输入图像，他们从重新缩放的训练集图像中随机裁剪（每个 SGD 迭代的每张图像进行一次裁剪）。增强训练集的方法：随机水平翻转和 RGB 颜色偏移。

Training image size. Let S be the smallest side of an isotropically-rescaled training image, from which the ConvNet input is cropped (we also refer to S as the training scale). While the crop size is fixed to 224×224, in principle S can take on any value not less than 224: for $S = 224$ the crop will capture whole-image statistics, completely spanning the smallest side of a training image; for $S \gg 224$ the crop will correspond to a small part of the image, containing a small object or an object part.

设 S 是各向同性重新缩放的训练图像的最小一侧长度（也称 S 为训练尺度）。
虽然剪裁大小固定为 224×224，但原则上 S 可以取不小于 224 的任何值：

$S = 224$，剪裁将捕获整幅图像统计数据，完全跨越训练图像的最小侧;
$S \gg 224$，剪裁将对应于图像的一小部分，包含一个小物体或物体的一部分。
部分。

We consider two approaches for setting the training scale S. The first is to fix S, which corresponds to single-scale training (note that image content within the sampled crops can still represent multi-scale image statistics). In our experiments, we evaluated models trained at two fixed scales: $S = 256$ (which has been widely used in the prior art (Krizhevsky et al., 2012; Zeiler & Fergus, 2013; Sermanet et al., 2014)) and $S = 384$. Given a ConvNet configuration, we first trained the network using $S = 256$. To speed-up training of the $S = 384$ network, it was initialised with the weights pre-trained with $S = 256$, and we used a smaller initial learning rate of 10^{-3}.

The second approach to setting S is multi-scale training, where each training image is individually rescaled by randomly sampling S from a certain range $[S_{min}, S_{max}]$ (we used $S_{min} = 256$ and $S_{max} = 512$). Since objects in images can be of different size, it is beneficial to take this into account during training. This can also be seen as training set augmentation by scale jittering, where a single model is trained to recognise objects over a wide range of scales. For speed reasons, we trained multi-scale models by fine-tuning all layers of a single-scale model with the same configuration, pre-trained with fixed $S = 384$.

我们考虑设置训练尺度 S 的两种方法：
(1) 使用固定的输入图片尺寸 S
论文评估了以两个固定尺度训练的模型：S=256 和 S=384
给定一个卷积神经网络配置，我们首先使用 S=256 来训练网络。为了加速 S=384 的训练，网络权重被初始化为 S=256 的预先训练的权重，并且使用 0.001 的较小的初始学习速率。
(2) 使用变长的输入尺寸 S，$S \in [S_{min}, S_{max}]$
每个训练图像通过在区间内随机取样 S 而单独重新缩放。
由于图像中的物体可能具有不同的大小，因此在训练中考虑这一点是有益的。这也可以看作是通过缩放抖动来增强训练集，其中单个模型被训练以识别范围广泛的物体。出于速度的原因，我们通过使用相同配置对单尺寸模型的所有图层进行微调来训练多尺度模型，预先训练使用固定的尺寸 S=384

3.2 Testing

At test time, given a trained ConvNet and an input image, it is classified in the following way. First, it is isotropically rescaled to a pre-defined smallest image side, denoted as Q (we also refer to it as the test scale). We note that Q is not necessarily equal to the training scale S (as we will show in Sect. 4, using several values of Q for each S leads to improved performance). Then, the network is applied densely over the rescaled test image in a way similar to (Sermanet et al., 2014). Namely, the fully-connected layers are first converted to convolutional layers (the first FC layer to a 7×7 conv. layer, the last two FC layers to 1×1 conv. layers). The resulting fully-convolutional net is then applied to the whole (uncropped) image. The result is a class score map with the number of channels equal to the number of classes, and a variable spatial resolution, dependent on the input image size. Finally, to obtain a fixed-size vector of class scores for the image, the class score map is spatially averaged (sum-pooled). We also augment the test set by horizontal flipping of the images; the soft-max class posteriors of the original and flipped images are averaged to obtain the final scores for the image.

首先将输入图像各向同性地缩放到预定义的最小图像边长，表示为 Q（测试尺度）。
Q 不一定等于 S。
完全连接层首先被转换成卷积层，将所得到的全卷积网络应用于整个（未裁剪的）图像。
结果是一个分类分数映射，通道数量等于分类数量以及一个可变的空间分辨率

（取决于输入图像的大小）
为获得图像分类分数的固定大小向量，分类分数映射是空间平均的（sum-pooled）我们还通过水平翻转图像来增加测试集;对原始图像和翻转图像的 softmax 类后代进行平均以获得图像的最终得分。（多个输入在 softmax 层做平均输出）

关于这一段的解释：
在测试阶段中，作者把测试图片的最短边设为 Q(Q 和训练过程中图片的最短边 S 不一定相等)，而且可以对同一张测试图片 rescale 成不同的大小，即多个 scale 的同一张图片都在网络中进行测试，这样可以提升测试效果。在测试阶段，作者参考 Sermanet 等人的做法，把网络的连接层转换为多个卷积层进行处理。
（可以把全连接层中的每个节点看作是1×1大小的 feature map，若有 n 个节点，看作是1×1的 n 通道的 feature maps）
以 VGG-16 为例：对于$224 \times 224 \times 3$的输入，最后一层卷积可得输出为$7 \times 7 \times 512$，如后层是一层含 4096 个神经元的全连接层，则可用卷积核为$7 \times 7 \times 512 \times 4096$的全局卷积来实现这一全连接运算过程。

Since the fully-convolutional network is applied over the whole image, there is no need to sample multiple crops at test time (Krizhevsky et al., 2012), which is less efficient as it requires network re-computation for each crop. At the same time, using a large set of crops, as done by Szegedy et al. (2014), can lead to improved accuracy, as it results in a finer sampling of the input image compared to the fully-convolutional net. Also, multi-crop evaluation is complementary to dense evaluation due to different convolution boundary conditions: when applying a ConvNet to a crop, the convolved feature maps are padded with zeros, while in the case of dense evaluation the padding for the same crop naturally comes from the neighbouring parts of an image (due to both the convolutions and spatial pooling), which substantially increases the overall network receptive field, so more context is captured. While we believe that in practice the increased computation time of multiple crops does not justify the potential gains in accuracy, for reference we also evaluate our networks using 50 crops per scale (5×5 regular grid with 2 flips), for a total of 150 crops over 3 scales, which is comparable to 144 crops over 4 scales used by Szegedy et al. (2014).

全卷积网络应用于整个图片之后，不需要像 AlexNet 中那样对一个测试图片进行多次裁剪（低效，需要网络为每个 crop 重新计算一遍）
AlexNet 需要对不同 crop 分别进行计算，而 VGG 中可以一次计算获取同一张图片不同区域的分类结果。
同时，把图片切成大量的 crop，可以提高准确率。
在做卷积操作时，边界做零扩展，而在 multi-crop 评估的情况下，图像扩展来自其相邻部分。

3.3 Implementation Details

Our implementation is derived from the publicly available C++ Caffe toolbox (Jia, 2013) (branched out in December 2013), but contains a number of significant modifications, allowing us to perform training and evaluation on multiple GPUs installed in a single system, as well as train and evaluate on full-size (uncropped) images at multiple scales (as described above). Multi-GPU training exploits data parallelism, and is carried out by splitting each batch of training images into several GPU batches, processed in parallel on each GPU. After the GPU batch gradients are computed, they are averaged to obtain the gradient of the full batch. Gradient computation is synchronous across the GPUs, so the result is exactly the same as when training on a single GPU.

While more sophisticated methods of speeding up ConvNet training have been recently proposed (Krizhevsky, 2014), which employ model and data parallelism for different layers of the net, we have found that our conceptually much simpler scheme already provides a speedup of 3.75 times on an off-the-shelf 4-GPU system, as compared to using a single GPU. On a system equipped with four NVIDIA Titan Black GPUs, training a single net took 2–3 weeks depending on the architecture.

实现基于 caffe，做了一定的修改，允许在单系统的多核 cpu 上进行训练，用多 GPU 并行计算训练图像的每个 batch，当所有 GPU 都计算完成之后，求所有 batch 得到梯度的平均值。

4. Classification Experiments（分类实验）

4.1 Single Scale Evaluation

单一尺度，Q 值固定

We begin with evaluating the performance of individual ConvNet models at a single scale with the layer configurations described in Sect. 2.2. The test image size was set as follows: $Q = S$ for fixed S, and $Q = 0.5(S_{min} + S_{max})$ for jittered $S \in [S_{min}, S_{max}]$. The results of are shown in Table 3.

Table 3: **ConvNet performance at a single test scale.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
A	256	256	29.6	10.4
A-LRN	256	256	29.7	10.5
B	256	256	28.7	9.9
C	256	256	28.1	9.4
	384	384	28.1	9.3
	[256;512]	384	27.3	8.8
D	256	256	27.0	8.8
	384	384	26.8	8.7
	[256;512]	384	25.6	8.1
E	256	256	27.3	9.0
	384	384	26.9	8.7
	[256;512]	384	**25.5**	**8.0**

结论：scale jittering 下有更低的错误率。

4.2 Multi-scale Evaluation

取多个 Q 值

Having evaluated the ConvNet models at a single scale, we now assess the effect of scale jittering at test time. It consists of running a model over several rescaled versions of a test image (corresponding to different values of Q), followed by averaging the resulting class posteriors. Considering that a large discrepancy between training and testing scales leads to a drop in performance, the models trained with fixed S were evaluated over three test image sizes, close to the training one: $Q = \{S - 32, S, S + 32\}$. At the same time, scale jittering at training time allows the network to be applied to a wider range of scales at test time, so the model trained with variable $S \in [S_{min}; S_{max}]$ was evaluated over a larger range of sizes $Q = \{S_{min}, 0.5(S_{min} + S_{max}), S_{max}\}$.

考虑到由训练和测试的尺度不同导致性能上的巨大差异：模型使用固定的训练输入尺寸 S 来评估三种尺寸 的测试图片，即测试输入图片尺寸 Q 取 $\{S-32, S, S+32\}$。 同时，我们可以对训练尺寸 S 做尺寸抖动，抖动范围为 $S \in [Smin; Smax]$，相对的测试图片的大小就变为 $Q = \{Smin, 0.5(Smin + Smax), Smax\}$. 在表格中，我们可以看见带尺寸抖动的模型是要优于不带抖动的。

Table 4: **ConvNet performance at multiple test scales.**

ConvNet config. (Table 1)	smallest image side		top-1 val. error (%)	top-5 val. error (%)
	train (S)	test (Q)		
B	256	224,256,288	28.2	9.6
C	256	224,256,288	27.7	9.2
	384	352,384,416	27.8	9.2
	[256; 512]	256,384,512	26.3	8.2
D	256	224,256,288	26.6	8.6
	384	352,384,416	26.5	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**
E	256	224,256,288	26.9	8.7
	384	352,384,416	26.7	8.6
	[256; 512]	256,384,512	**24.8**	**7.5**

结论：多尺度测试比单尺度结果要好，且使用 scale jittering 有助于提高模型的性能。

4.3 Multi-crop Evaluation

In Table 5 we compare dense ConvNet evaluation with mult-crop evaluation (see Sect. 3.2 for details). We also assess the complementarity of the two evaluation techniques by averaging their soft-max outputs. As can be seen, using multiple crops performs slightly better than dense evaluation, and the two approaches are indeed complementary, as their combination outperforms each of them. As noted above, we hypothesize that this is due to a different treatment of convolution boundary conditions.

Table 5: **ConvNet evaluation techniques comparison.** In all experiments the training scale S was sampled from $[256; 512]$, and three test scales Q were considered: $\{256, 384, 512\}$.

ConvNet config. (Table 1)	Evaluation method	top-1 val. error (%)	top-5 val. error (%)
D	dense	24.8	7.5
	multi-crop	24.6	7.5
	multi-crop & dense	**24.4**	**7.2**
E	dense	24.8	7.5
	multi-crop	24.6	7.4
	multi-crop & dense	**24.4**	**7.1**

结论：单独使用 multi-crop evaluation 要比单独使用 dense evaluation 效果好，两个方法同时使用时，要比单独使用任一种都好。

4.4 ConvNet Fusion（融合）

Up until now, we evaluated the performance of individual ConvNet models. In this part of the experiments, we combine the outputs of several models by averaging their soft-max class posteriors. This improves the performance due to complementarity of the models, and was used in the top ILSVRC submissions in 2012 (Krizhevsky et al., 2012) and 2013 (Zeiler & Fergus, 2013; Sermanet et al., 2014).

结论：将几个模型的 soft-max 分类策略的输出求平均后再用于识别，这样可以提高最后的性能。

4.5 Comparison with the State of the Art

Table 7: **Comparison with the state of the art in ILSVRC classification**. Our method is denoted as "VGG". Only the results obtained without outside training data are reported.

Method	top-1 val. error (%)	top-5 val. error (%)	top-5 test error (%)
VGG (2 nets, multi-crop & dense eval.)	**23.7**	**6.8**	**6.8**
VGG (1 net, multi-crop & dense eval.)	24.4	7.1	7.0
VGG (ILSVRC submission, 7 nets, dense eval.)	24.7	7.5	7.3
GoogLeNet (Szegedy et al., 2014) (1 net)	-	7.9	
GoogLeNet (Szegedy et al., 2014) (7 nets)	-	**6.7**	
MSRA (He et al., 2014) (11 nets)	-	-	8.1
MSRA (He et al., 2014) (1 net)	27.9	9.1	9.1
Clarifai (Russakovsky et al., 2014) (multiple nets)	-	-	11.7
Clarifai (Russakovsky et al., 2014) (1 net)	-	-	12.5
Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)	36.0	14.7	14.8
Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)	37.5	16.0	16.1
OverFeat (Sermanet et al., 2014) (7 nets)	34.0	13.2	13.6
OverFeat (Sermanet et al., 2014) (1 net)	35.7	14.2	-
Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)	38.1	16.4	16.4
Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)	40.7	18.2	-

结论：挺好

5. Conclusion（总结）

我们评估了非常深的卷积网络（多达 19 个权重层）用于大规模图像分类。已经证明，增加深度有利于分类准确性的提升，通过大幅增加深度，可以使用传统的 ConvNet 架构实现更好的性能。

附录中，展示了模型应用于更广泛的任务和数据集的情况，可以和围绕深度图像表示形成的更复杂的识别流水线的结果相当或者更优。

综上：实验的结果证实了深度在卷积神经网络中的重要性。

A Localisation

A.1 Localisation ConvNet

A.2 Localisation Experiments

B Generalisation of Very Deep Features