Going Deeper with Convolutions

Christian Szegedy[1], Wei Liu[2], Yangqing Jia[1], Pierre Sermanet[1], Scott Reed[3],
Dragomir Anguelov[1], Dumitru Erhan[1], Vincent Vanhoucke[1], Andrew Rabinovich[4]
[1]Google Inc. [2]University of North Carolina, Chapel Hill
[3]University of Michigan, Ann Arbor [4]Magic Leap Inc.
[1]{szegedy,jiayq,sermanet,dragomir,dumitru,vanhoucke}@google.com
[2]wliu@cs.unc.edu, [3]reedscott@umich.edu, [4]arabinovich@magicleap.com

Abstract

We propose a deep convolutional neural network architecture codenamed Inception that achieves the new state of the art for classification and detection in the ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC14). The main hallmark of this architecture is the improved utilization of the computing resources inside the network. By a carefully crafted design, we increased the depth and width of the network while keeping the computational budget constant. To optimize quality, the architectural decisions were based on the Hebbian principle and the intuition of multi-scale processing. One particular incarnation used in our submission for ILSVRC14 is called GoogLeNet, a 22 layers deep network, the quality of which is assessed in the context of classification and detection.

1. Introduction

In the last three years, our object classification and detection capabilities have dramatically improved due to advances in deep learning and convolutional networks [10]. One encouraging news is that most of this progress is not just the result of more powerful hardware, larger datasets and bigger models, but mainly a consequence of new ideas, algorithms and improved network architectures. No new data sources were used, for example, by the top entries in the ILSVRC 2014 competition besides the classification dataset of the same competition for detection purposes. Our GoogLeNet submission to ILSVRC 2014 actually uses 12 times fewer parameters than the winning architecture of Krizhevsky et al [9] from two years ago, while being significantly more accurate. On the object detection front, the biggest gains have not come from naive application of bigger and bigger deep networks, but from the synergy of deep architectures and classical computer vision, like the R-CNN algorithm by Girshick et al [6].

GoogleNet比ImageNet使用的参数少12倍，但是准确率更高。依赖的不是一味地增加网络的深度和宽度，而是类似R-CNN算法的深度架构与经典计算机视觉的协同效应。

Another notable factor is that with the ongoing traction of mobile and embedded computing, the efficiency of our algorithms – especially their power and memory use – gains importance. It is noteworthy that the considerations leading to the design of the deep architecture presented in this paper included this factor rather than having a sheer fixation on accuracy numbers. For most of the experiments, the models were designed to keep a computational budget of 1.5 billion multiply-adds at inference time, so that the they do not end up to be a purely academic curiosity, but could be put to real world use, even on large datasets, at a reasonable cost.

手机和电脑硬件发展，使得神经网络成为可能。
不限于学术，此模型在预测阶段只有15亿次加乘操作，可用于现实。

In this paper, we will focus on an efficient deep neural network architecture for computer vision, codenamed Inception, which derives its name from the Network in network paper by Lin et al [12] in conjunction with the famous “we need to go deeper” internet meme [1]. In our case, the word “deep” is used in two different meanings: first of all, in the sense that we introduce a new level of organization in the form of the “Inception module” and also in the more direct sense of increased network depth. In general, one can view the Inception model as a logical culmination of [12] while taking inspiration and guidance from the theoretical work by Arora et al [2]. The benefits of the architecture are experimentally verified on the ILSVRC 2014 classification and detection challenges, where it significantly outperforms the current state of the art.

2. Related Work

Starting with LeNet-5 [10], convolutional neural networks (CNN) have typically had a standard structure – stacked convolutional layers (optionally followed by con-

卷积神经网络有标准的架构，卷积层之后是一或多层全连接层。 这些基本设计出来的网络在图像识别领域取得了最好的结果。

近年来的趋势是增加卷积层数和卷积层大小，使用dropout防止过拟合问题

trast normalization and max-pooling) are followed by one or more fully-connected layers. Variants of this basic design are prevalent in the image classification literature and have yielded the best results to-date on MNIST, CIFAR and most notably on the ImageNet classification challenge [9, 21]. For larger datasets such as Imagenet, the recent trend has been to increase the number of layers [12] and layer size [21, 14], while using dropout [7] to address the problem of overfitting.

尽管最大池化会丢失空间信息，相同的卷积神经网络成功用于物体识别，人类手势识别

Despite concerns that max-pooling layers result in loss of accurate spatial information, the same convolutional network architecture as [9] has also been successfully employed for localization [9, 14], object detection [6, 14, 18, 5] and human pose estimation [19].

论文15使用一系列不同尺寸的固定Gabor滤波器来处理不同尺寸问题。

Inspired by a neuroscience model of the primate visual cortex, Serre et al. [15] used a series of fixed Gabor filters of different sizes to handle multiple scales. We use a similar strategy here. However, contrary to the fixed 2-layer deep model of [15], all filters in the Inception architecture are learned. Furthermore, Inception layers are repeated many times, leading to a 22-layer deep model in the case of the GoogLeNet model.

GoogleNet所有层都会学习这种策略，并重复

网络中的网络可用来增加NN的表现力。此模型中1x1卷积层被大量使用。双重目的：降维的同时增加网络深度和宽度

Network-in-Network is an approach proposed by Lin et al. [12] in order to increase the representational power of neural networks. In their model, additional 1×1 convolutional layers are added to the network, increasing its depth. We use this approach heavily in our architecture. However, in our setting, 1×1 convolutions have dual purpose: most critically, they are used mainly as dimension reduction modules to remove computational bottlenecks, that would otherwise limit the size of our networks. This allows for not just increasing the depth, but also the width of our networks without a significant performance penalty.

区域CNN将物体识别问题分解为两个子问题。先利用低级线索如颜色和文本生成物体位置，然后使用CNN分类器分类。

Finally, the current state of the art for object detection is the Regions with Convolutional Neural Networks (R-CNN) method by Girshick et al. [6]. R-CNN decomposes the overall detection problem into two subproblems: utilizing low-level cues such as color and texture in order to generate object location proposals in a category-agnostic fashion and using CNN classifiers to identify object categories at those locations. Such a two stage approach leverages the accuracy of bounding box segmentation with low-level cues, as well as the highly powerful classification power of state-of-the-art CNNs. We adopted a similar pipeline in our detection submissions, but have explored enhancements in both stages, such as multi-box [5] prediction for higher object bounding box recall, and ensemble approaches for better categorization of bounding box proposals.

3. Motivation and High Level Considerations

The most straightforward way of improving the performance of deep neural networks is by increasing their size. This includes both increasing the depth – the number of network levels – as well as its width: the number of units at each level. This is an easy and safe way of training higher quality models, especially given the availability of a large amount of labeled training data. However, this simple solution comes with two major drawbacks.

提升深度神经网络的性能的最直接的方法就是增加网络的深度和广度，但是这两种方法有两个短板。

Figure 1: Two distinct classes from the 1000 classes of the ILSVRC 2014 classification challenge. Domain knowledge is required to distinguish between these classes.

Bigger size typically means a larger number of parameters, which makes the enlarged network more prone to overfitting, especially if the number of labeled examples in the training set is limited. This is a major bottleneck as strongly labeled datasets are laborious and expensive to obtain, often requiring expert human raters to distinguish between various fine-grained visual categories such as those in ImageNet (even in the 1000-class ILSVRC subset) as shown in Figure 1.

网络尺寸越大，参数越多，越容易过拟合，尤其当网络标签数目有限时。

The other drawback of uniformly increased network size is the dramatically increased use of computational resources. For example, in a deep vision network, if two convolutional layers are chained, any uniform increase in the number of their filters results in a quadratic increase of computation. If the added capacity is used inefficiently (for example, if most weights end up to be close to zero), then much of the computation is wasted. As the computational budget is always finite, an efficient distribution of computing resources is preferred to an indiscriminate increase of size, even when the main objective is to increase the quality of performance.

另一个问题是，统一增加网络尺寸将极大增加计算消耗。若两个卷积层是链式，增加任何一个的filter的数量将导致计算消耗二次增加。如果增加的层没有正确应用（比如最后权值衰减为0），只会消耗有限的计算资源。

A fundamental way of solving both of these issues would be to introduce sparsity and replace the fully connected layers by the sparse ones, even inside the convolutions. Besides mimicking biological systems, this would also have the advantage of firmer theoretical underpinnings due to the groundbreaking work of Arora et al. [2]. Their main result states that if the probability distribution of the dataset is representable by a large, very sparse deep neural network, then the optimal network topology can be constructed layer after layer by analyzing the correlation statistics of the preceding layer activations and clustering neurons with highly correlated outputs. Although the strict mathematical proof requires very strong conditions, the fact that this statement

一种基础的解决方案是引入稀疏层并替换全连接层，甚至在卷积层内部。

某篇论文证明

如果数据集的概率分布可以用很大很稀疏的深度神经网络表示，那么优化网络拓扑图可以通过分析前馈层激活函数和高度相关的神经元集群输出的相关统计特性来一层层构建。

其实，这就是著名的Hebbian定律。

resonates with the well known Hebbian principle – neurons that fire together, wire together – suggests that the underlying idea is applicable even under less strict conditions, in practice.

当今的计算设施面临非均匀稀疏的数据结构依然乏力。即便算术操作减少到1%

Unfortunately, today's computing infrastructures are very inefficient when it comes to numerical calculation on non-uniform sparse data structures. Even if the number of arithmetic operations is reduced by $100\times$, the overhead of lookups and cache misses would dominate: switching to sparse matrices might not pay off. The gap is widened yet further by the use of steadily improving and highly tuned numerical libraries that allow for extremely fast dense matrix multiplication, exploiting the minute details of the underlying CPU or GPU hardware [16, 9]. Also, non-uniform sparse models require more sophisticated engineering and computing infrastructure. Most current vision oriented machine learning systems utilize sparsity in the spatial domain just by the virtue of employing convolutions. However, convolutions are implemented as collections of dense connections to the patches in the earlier layer. ConvNets have traditionally used random and sparse connection tables in the feature dimensions since [11] in order to break the symmetry and improve learning, yet the trend changed back to full connections with [9] in order to further optimize parallel computation. Current state-of-the-art architectures for computer vision have uniform structure. The large number of filters and greater batch size allows for the efficient use of dense computation.

This raises the question of whether there is any hope for a next, intermediate step: an architecture that makes use of filter-level sparsity, as suggested by the theory, but exploits our current hardware by utilizing computations on dense matrices. The vast literature on sparse matrix computations (e.g. [3]) suggests that clustering sparse matrices into relatively dense submatrices tends to give competitive performance for sparse matrix multiplication. It does not seem far-fetched to think that similar methods would be utilized for the automated construction of non-uniform deep-learning architectures in the near future.

GoogleNet的架构始于一次研究，一个试图拟合由论文2实现的用于处理视觉任务的稀疏的复杂网络结构的假设输出

The Inception architecture started out as a case study for assessing the hypothetical output of a sophisticated network topology construction algorithm that tries to approximate a sparse structure implied by [2] for vision networks and covering the hypothesized outcome by dense, readily available components. Despite being a highly speculative undertaking, modest gains were observed early on when compared with reference networks based on [12]. With a bit of tuning the gap widened and Inception proved to be especially useful in the context of localization and object detection as the base network for [6] and [5]. Interestingly, while most of the original architectural choices have been questioned and tested thoroughly in separation, they turned out to be close to optimal locally. One must be cautious though: although the Inception architecture has become a success for computer vision, it is still questionable whether this can be attributed to the guiding principles that have lead to its construction. Making sure of this would require a much more thorough analysis and verification.

4. Architectural Details

googlenet架构的主要思想是考虑一个卷积视觉网络内的优化局部稀疏结构如何被一些稳定可用的稠密组件拟合。

The main idea of the Inception architecture is to consider how an optimal local sparse structure of a convolutional vision network can be approximated and covered by readily available dense components. Note that assuming translation invariance means that our network will be built from convolutional building blocks. All we need is to find the optimal local construction and to repeat it spatially. Arora et al. [2] suggests a layer-by layer construction where one should analyze the correlation statistics of the last layer and cluster them into groups of units with high correlation. These clusters form the units of the next layer and are connected to the units in the previous layer. We assume that each unit from an earlier layer corresponds to some region of the input image and these units are grouped into filter banks. In the lower layers (the ones close to the input) correlated units would concentrate in local regions. Thus, we would end up with a lot of clusters concentrated in a single region and they can be covered by a layer of 1×1 convolutions in the next layer, as suggested in [12]. However, one can also expect that there will be a smaller number of more spatially spread out clusters that can be covered by convolutions over larger patches, and there will be a decreasing number of patches over larger and larger regions. In order to avoid patch-alignment issues, current incarnations of the Inception architecture are restricted to filter sizes 1×1, 3×3 and 5×5; this decision was based more on convenience rather than necessity. It also means that the suggested architecture is a combination of all those layers with their output filter banks concatenated into a single output vector forming the input of the next stage. Additionally, since pooling operations have been essential for the success of current convolutional networks, it suggests that adding an alternative parallel pooling path in each such stage should have additional beneficial effect, too (see Figure 2(a)).

注意到，转换不变性意味着网络可以通过卷积构建块来构建。我们需要做的是找到优化的局部结构并从空间中重复。

论文2提到的一层层地构建网络可以通过分析最后一层的相关统计特性，并将它们中高度相关的聚集为组。这些集群形成了下一层的基本单元并与上一层的单元相连。假设前些层的单元对应了输入图片的某些区域，这些单元聚集为filters库。靠近输入的层的相关单元会集中为局部区域。因此，我们会得到许多集中为单个区域并可以被1x1卷积层覆盖的集群。

为避免patch对齐问题，当前网络架构变换都被限制为filter size 1X1,3X3,5X5,仅仅为了方便而非必要。

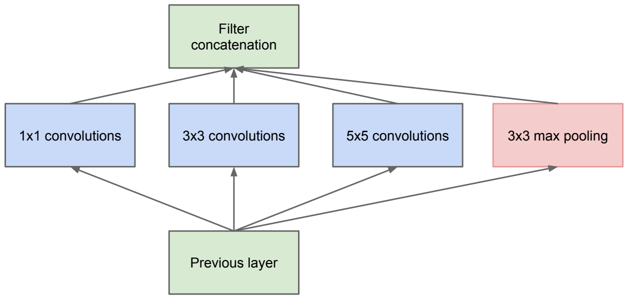
除此之外，每一个这样的阶段增加并行的池化操作有额外的好处。

层数越高，3x3和5x5卷积比例越大。

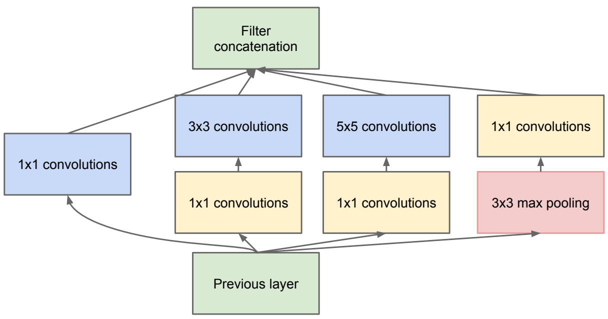
As these "Inception modules" are stacked on top of each other, their output correlation statistics are bound to vary: as features of higher abstraction are captured by higher layers, their spatial concentration is expected to decrease. This suggests that the ratio of 3×3 and 5×5 convolutions should increase as we move to higher layers.

One big problem with the above modules, at least in this naïve form, is that even a modest number of 5×5 convolutions can be prohibitively expensive on top of a convolutional layer with a large number of filters. This problem becomes even more pronounced once pooling units are added to the mix: the number of output filters equals to the num-

有个大问题是，即便5x5这样中和的卷基层，如果置于包含大量filter的卷积之上也会是计算消耗极大的。一旦池化单元混入，这个问题将变得更加尖锐，filter的输出数量等于前一层filter的数量。



(a) Inception module, naïve version



(b) Inception module with dimensionality reduction

Figure 2: Inception module

池化层的输出和卷基层输出的聚合将不可避免的随阶段递增而增加输出数量。然而，我们的架构可能覆盖了优化的稀疏结构，这样会非常低效，并导致计算的飞速膨胀。

ber of filters in the previous stage. The merging of output of the pooling layer with outputs of the convolutional layers would lead to an inevitable increase in the number of outputs from stage to stage. While this architecture might cover the optimal sparse structure, it would do it very inefficiently, leading to a computational blow up within a few stages.

这引出googlenet的第二个idea，无论何时降维都有必要。

This leads to the second idea of the Inception architecture: judiciously reducing dimension wherever the computational requirements would increase too much otherwise. This is based on the success of embeddings: even low dimensional embeddings might contain a lot of information about a relatively large image patch. However, embeddings represent information in a dense, compressed form and compressed information is harder to process. The representation should be kept sparse at most places (as required by the conditions of [2]) and compress the signals only whenever they have to be aggregated en masse. That is, 1×1 convolutions are used to compute reductions before the expensive 3×3 and 5×5 convolutions. Besides being used as reductions, they also include the use of rectified linear activation making them dual-purpose. The final result is depicted in Figure 2(b).

embeding represent 信息到一个压缩稠密的形式是很难处理的。

representation应该在大部分地方稀疏并在信号聚集的地方压缩。

这即，1x1的卷积用于3x3,5x5卷积之前的计算缩减。除此之外，1x1还用于线性整流激活Relu

In general, an Inception network is a network consisting of modules of the above type stacked upon each other, with occasional max-pooling layers with stride 2 to halve the resolution of the grid. For technical reasons (memory efficiency during training), it seemed beneficial to start using Inception modules only at higher layers while keeping the lower layers in traditional convolutional fashion. This is not strictly necessary, simply reflecting some infrastructural inefficiencies in our current implementation.

更普遍的讲，一个Inceptions网络是一个由以上模块，随机添加stride=2的以保证网格分辨率的最大池化层相互堆积而成

A useful aspect of this architecture is that it allows for increasing the number of units at each stage significantly without an uncontrolled blow-up in computational complexity at later stages. This is achieved by the ubiquitous use of dimensionality reduction prior to expensive convolutions with larger patch sizes. Furthermore, the design follows the practical intuition that visual information should be processed at various scales and then aggregated so that the next stage can abstract features from the different scales simultaneously.

此架构的裨益在于，允许随意增加每一阶段的单元数而不至于让计算消耗极剧增加。这得益于先于计算消耗较大的卷积的使用较大patch size的维度缩减

The improved use of computational resources allows for increasing both the width of each stage as well as the number of stages without getting into computational difficulties. One can utilize the Inception architecture to create slightly inferior, but computationally cheaper versions of it. We have found that all the available knobs and levers allow for a controlled balancing of computational resources resulting in networks that are $3-10\times$ faster than similarly performing networks with non-Inception architecture, however this requires careful manual design at this point.

我们发现了不使用Inception架构的所有可用的开关和杠杆，用以控制计算资源以3-10倍计算速度获得相同网络性能

5. GoogLeNet

By the"GoogLeNet" name we refer to the particular incarnation of the Inception architecture used in our submission for the ILSVRC 2014 competition. We also used one deeper and wider Inception network with slightly superior quality, but adding it to the ensemble seemed to improve the results only marginally. We omit the details of that network, as empirical evidence suggests that the influence of the exact architectural parameters is relatively minor. Table 1 illustrates the most common instance of Inception used in the competition. This network (trained with different image-patch sampling methods) was used for 6 out of the 7 models in our ensemble.

我们使用一个更深更广的Inception网络，但是将其加入集合所带来的提升微乎其微。我们舍弃inception网络细节，因为经验表明抽取参数相对次要。

All the convolutions, including those inside the Inception modules, use rectified linear activation. The size of the receptive field in our network is 224×224 in the RGB color space with zero mean. "#3×3 reduce" and "#5×5 reduce" stands for the number of 1×1 filters in the reduction layer used before the 3×3 and 5×5 convolutions. One can see the number of 1×1 filters in the projection layer after the built-in max-pooling in the pool proj column. All these reduction/projection layers use rectified linear activation as well.

所有的卷积层，包括inception模型内部的，使用Relu激活函数。网络接收范围是224x224的RGB颜色空间，均值为0.

"#3X3 reduce"和"#5x5 reduce"代表3x3和5x5卷积层之前的降维的1x1 filter 数量。可以在pool proj列内建最大池化之后看到1x1filter的数量。所有的这些 reduction/projection层都使用Relu

The network was designed with computational efficiency and practicality in mind, so that inference can be run on individual devices including even those with limited computational resources, especially with low-memory footprint.

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

Table 1: GoogLeNet incarnation of the Inception architecture.

如果只计算包含参数的，网络有22层，若包含池化层有27层。独立单元有100个左右。

在分类之前使用均值池化基于论文12。我们的实现有一层额外的线性层，这一线性层使得网络有能力适应其他标签集。将全连接层换成均值池化可以将top-1准确率提升0.6%，但在移除全连接层时使用dropout层仍然重要

The network is 22 layers deep when counting only layers with parameters (or 27 layers if we also count pooling). The overall number of layers (independent building blocks) used for the construction of the network is about 100. The exact number depends on how layers are counted by the machine learning infrastructure. The use of average pooling before the classifier is based on [12], although our implementation has an additional linear layer. The linear layer enables us to easily adapt our networks to other label sets, however it is used mostly for convenience and we do not expect it to have a major effect. We found that a move from fully connected layers to average pooling improved the top-1 accuracy by about 0.6%, however the use of dropout remained essential even after removing the fully connected layers.

Given relatively large depth of the network, the ability to propagate gradients back through all the layers in an effective manner was a concern. The strong performance of shallower networks on this task suggests that the features produced by the layers in the middle of the network should be very discriminative. By adding auxiliary classifiers connected to these intermediate layers, discrimination in the lower stages in the classifier was expected. This was thought to combat the vanishing gradient problem while providing regularization. These classifiers take the form of smaller convolutional networks put on top of the output of the Inception (4a) and (4d) modules. During training, their loss gets added to the total loss of the network with a discount weight (the losses of the auxiliary classifiers were weighted by 0.3). At inference time, these auxiliary networks are discarded. Later control experiments have shown that the effect of the auxiliary networks is relatively minor (around 0.5%) and that it required only one of them to achieve the same effect.

鉴于网络较大，其反向传播能力需要考虑。此任务的浅层网络的表现表明网络中间层产生的特征尤为重要。
通过在连接到这些中间层上增加辅助分类器

这些分类器以小型卷积神经网络的形式放置于inception(4a)和模块(4d)的输出之上。
网络训练期间，它们的loss会以一定比率权重累加到网络总loss中。
预测阶段，这些分类器都会被舍弃。后续的控制实验表明，辅助网络的效果甚微（0.5%左右）

The exact structure of the extra network on the side, including the auxiliary classifier, is as follows:

辅助网络的精确结构如下：

- An average pooling layer with 5×5 filter size and stride 3, resulting in an $4\times4\times512$ output for the (4a), and $4\times4\times528$ for the (4d) stage.
- A 1×1 convolution with 128 filters for dimension reduction and rectified linear activation.
- A fully connected layer with 1024 units and rectified linear activation.
- A dropout layer with 70% ratio of dropped outputs.

- A linear layer with softmax loss as the classifier (predicting the same 1000 classes as the main classifier, but removed at inference time).

A schematic view of the resulting network is depicted in Figure 3.

6. Training Methodology

GoogLeNet networks were trained using the DistBelief [4] distributed machine learning system using modest amount of model and data-parallelism. Although we used a CPU based implementation only, a rough estimate suggests that the GoogLeNet network could be trained to convergence using few high-end GPUs within a week, the main limitation being the memory usage. Our training used asynchronous stochastic gradient descent with 0.9 momentum [17], fixed learning rate schedule (decreasing the learning rate by 4% every 8 epochs). Polyak averaging [13] was used to create the final model used at inference time.

Image sampling methods have changed substantially over the months leading to the competition, and already converged models were trained on with other options, sometimes in conjunction with changed hyperparameters, such as dropout and the learning rate. Therefore, it is hard to give a definitive guidance to the most effective single way to train these networks. To complicate matters further, some of the models were mainly trained on smaller relative crops, others on larger ones, inspired by [8]. Still, one prescription that was verified to work very well after the competition, includes sampling of various sized patches of the image whose size is distributed evenly between 8% and 100% of the image area with aspect ratio constrained to the interval $[\frac{3}{4}, \frac{4}{3}]$. Also, we found that the photometric distortions of Andrew Howard [8] were useful to combat overfitting to the imaging conditions of training data.

7. ILSVRC 2014 Classification Challenge Setup and Results

The ILSVRC 2014 classification challenge involves the task of classifying the image into one of 1000 leaf-node categories in the Imagenet hierarchy. There are about 1.2 million images for training, 50,000 for validation and 100,000 images for testing. Each image is associated with one ground truth category, and performance is measured based on the highest scoring classifier predictions. Two numbers are usually reported: the top-1 accuracy rate, which compares the ground truth against the first predicted class, and the top-5 error rate, which compares the ground truth against the first 5 predicted classes: an image is deemed correctly classified if the ground truth is among the top-5, regardless of its rank in them. The challenge uses the top-5 error rate for ranking purposes.

我们的训练方法中
使用异步更新的随机梯度
下降，动量为0.9,固定的
学习速率(每8个epochs
下降4%)

竞赛过程中调整了很多
，因而无法给出权威指导。

尽管如此，在竞赛之后一个方
法被证实很有用。
抽样图片的不同尺寸patched
，让其尺寸均衡分布在图像区域的
8%-100%之间，纵横比率限制在
间隔为[3/4,4/3]
同时，我们发现论文8提到的光
度扭曲对于训练数据中过拟合
问题很有效

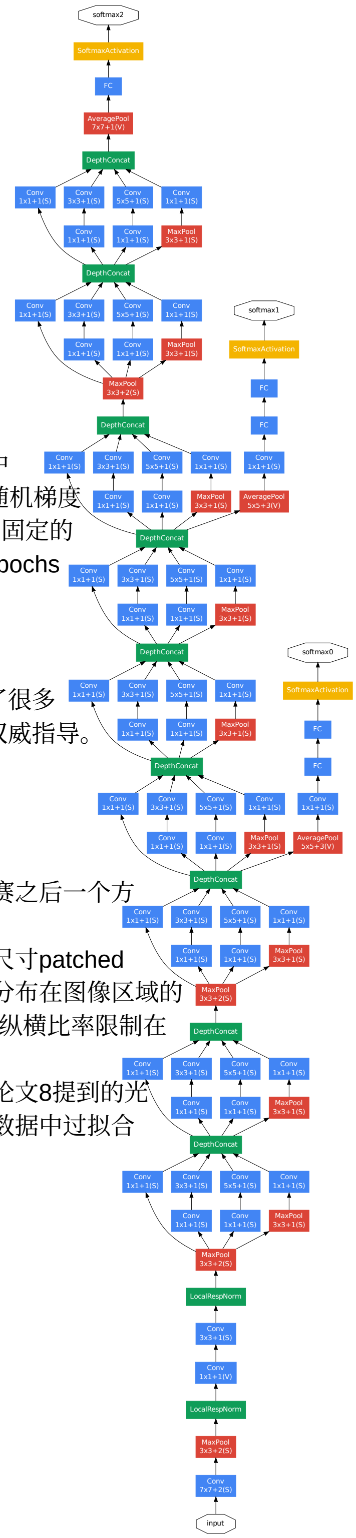


Figure 3: GoogLeNet network with all the bells and whistles.

We participated in the challenge with no external data used for training. In addition to the training techniques aforementioned in this paper, we adopted a set of techniques during testing to obtain a higher performance, which we describe next.

1. We independently trained 7 versions of the same GoogLeNet model (including one wider version), and performed ensemble prediction with them. These models were trained with the same initialization (even with the same initial weights, due to an oversight) and learning rate policies. They differed only in sampling methodologies and the randomized input image order.

2. During testing, we adopted a more aggressive cropping approach than that of Krizhevsky et al. [9]. Specifically, we resized the image to 4 scales where the shorter dimension (height or width) is 256, 288, 320 and 352 respectively, take the left, center and right square of these resized images (in the case of portrait images, we take the top, center and bottom squares). For each square, we then take the 4 corners and the center 224×224 crop as well as the square resized to 224×224, and their mirrored versions. This leads to $4\times3\times6\times2 = 144$ crops per image. A similar approach was used by Andrew Howard [8] in the previous year's entry, which we empirically verified to perform slightly worse than the proposed scheme. We note that such aggressive cropping may not be necessary in real applications, as the benefit of more crops becomes marginal after a reasonable number of crops are present (as we will show later on).

3. The softmax probabilities are averaged over multiple crops and over all the individual classifiers to obtain the final prediction. In our experiments we analyzed alternative approaches on the validation data, such as max pooling over crops and averaging over classifiers, but they lead to inferior performance than the simple averaging.

In the remainder of this paper, we analyze the multiple factors that contribute to the overall performance of the final submission.

Our final submission to the challenge obtains a top-5 error of 6.67% on both the validation and testing data, ranking the first among other participants. This is a 56.5% relative reduction compared to the SuperVision approach in 2012, and about 40% relative reduction compared to the previous year's best approach (Clarifai), both of which used external data for training the classifiers. Table 2 shows the statistics of some of the top-performing approaches over the past 3 years.

We also analyze and report the performance of multiple testing choices, by varying the number of models and the number of crops used when predicting an image in Table 3. When we use one model, we chose the one with the lowest top-1 error rate on the validation data. All numbers are reported on the validation dataset in order to not overfit to the testing data statistics.

Team	**Year**	**Place**	**Error (top-5)**	**Uses external data**
SuperVision	2012	1st	16.4%	no
SuperVision	2012	1st	15.3%	Imagenet 22k
Clarifai	2013	1st	11.7%	no
Clarifai	2013	1st	11.2%	Imagenet 22k
MSRA	2014	3rd	7.35%	no
VGG	2014	2nd	7.32%	no
GoogLeNet	2014	1st	6.67%	no

Table 2: Classification performance.

Number of models	**Number of Crops**	**Cost**	**Top-5 error**	**compared to base**
1	1	1	10.07%	base
1	10	10	9.15%	-0.92%
1	144	144	7.89%	-2.18%
7	1	7	8.09%	-1.98%
7	10	70	7.62%	-2.45%
7	144	1008	6.67%	-3.45%

Table 3: GoogLeNet classification performance break down.

8. ILSVRC 2014 Detection Challenge Setup and Results

The ILSVRC detection task is to produce bounding boxes around objects in images among 200 possible classes. Detected objects count as correct if they match the class of the groundtruth and their bounding boxes overlap by at least 50% (using the Jaccard index). Extraneous detections count as false positives and are penalized. Contrary to the classification task, each image may contain many objects or none, and their scale may vary. Results are reported using the mean average precision (mAP). The approach taken by GoogLeNet for detection is similar to the R-CNN by [6], but is augmented with the Inception model as the region classifier. Additionally, the region proposal step is improved by combining the selective search [20] approach with multibox [5] predictions for higher object bounding box recall. In order to reduce the number of false positives, the super-

Team	Year	Place	mAP	external data	ensemble	approach
UvA-Euvision	2013	1st	22.6%	none	?	Fisher vectors
Deep Insight	2014	3rd	40.5%	ImageNet 1k	3	CNN
CUHK DeepID-Net	2014	2nd	40.7%	ImageNet 1k	?	CNN
GoogLeNet	2014	1st	43.9%	ImageNet 1k	6	CNN

Table 4: Comparison of detection performances. Unreported values are noted with question marks.

pixel size was increased by $2\times$. This halves the proposals coming from the selective search algorithm. We added back 200 region proposals coming from multi-box [5] resulting, in total, in about 60% of the proposals used by [6], while increasing the coverage from 92% to 93%. The overall effect of cutting the number of proposals with increased coverage is a 1% improvement of the mean average precision for the single model case. Finally, we use an ensemble of 6 GoogLeNets when classifying each region. This leads to an increase in accuracy from 40% to 43.9%. Note that contrary to R-CNN, we did not use bounding box regression due to lack of time.

We first report the top detection results and show the progress since the first edition of the detection task. Compared to the 2013 result, the accuracy has almost doubled. The top performing teams all use convolutional networks. We report the official scores in Table 4 and common strategies for each team: the use of external data, ensemble models or contextual models. The external data is typically the ILSVRC12 classification data for pre-training a model that is later refined on the detection data. Some teams also mention the use of the localization data. Since a good portion of the localization task bounding boxes are not included in the detection dataset, one can pre-train a general bounding box regressor with this data the same way classification is used for pre-training. The GoogLeNet entry did not use the localization data for pretraining.

In Table 5, we compare results using a single model only. The top performing model is by Deep Insight and surprisingly only improves by 0.3 points with an ensemble of 3 models while the GoogLeNet obtains significantly stronger results with the ensemble.

Team	mAP	Contextual model	Bounding box regression
Trimps-Soushen	31.6%	no	?
Berkeley Vision	34.5%	no	yes
UvA-Euvision	35.4%	?	?
CUHK DeepID-Net2	37.7%	no	?
GoogLeNet	38.02%	no	no
Deep Insight	40.2%	yes	yes

Table 5: Single model performance for detection.

9. Conclusions

Our results yield a solid evidence that approximating the expected optimal sparse structure by readily available dense building blocks is a viable method for improving neural networks for computer vision. The main advantage of this method is a significant quality gain at a modest increase of computational requirements compared to shallower and narrower architectures.

Our object detection work was competitive despite not utilizing context nor performing bounding box regression, suggesting yet further evidence of the strengths of the Inception architecture.

For both classification and detection, it is expected that similar quality of result can be achieved by much more expensive non-Inception-type networks of similar depth and width. Still, our approach yields solid evidence that moving to sparser architectures is feasible and useful idea in general. This suggest future work towards creating sparser and more refined structures in automated ways on the basis of [2], as well as on applying the insights of the Inception architecture to other domains.

References

[1] Know your meme: We need to go deeper. `http://knowyourmeme.com/memes/we-need-to-go-deeper`. Accessed: 2014-09-15.

[2] S. Arora, A. Bhaskara, R. Ge, and T. Ma. Provable bounds for learning some deep representations. *CoRR*, abs/1310.6343, 2013.

[3] U. V. Çatalyürek, C. Aykanat, and B. Uçar. On two-dimensional sparse matrix partitioning: Models, methods, and a recipe. *SIAM J. Sci. Comput.*, 32(2):656–683, Feb. 2010.

[4] J. Dean, G. Corrado, R. Monga, K. Chen, M. Devin, M. Mao, M. Ranzato, A. Senior, P. Tucker, K. Yang, Q. V. Le, and A. Y. Ng. Large scale distributed deep networks. In P. Bartlett, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, editors, *NIPS*, pages 1232–1240. 2012.

[5] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014.

[6] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition, 2014. CVPR 2014. IEEE Conference on*, 2014.

[7] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012.

[8] A. G. Howard. Some improvements on deep convolutional neural network based image classification. *CoRR*, abs/1312.5402, 2013.

[9] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, 2012.

[10] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551, Dec. 1989.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] M. Lin, Q. Chen, and S. Yan. Network in network. *CoRR*, abs/1312.4400, 2013.

[13] B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM J. Control Optim.*, 30(4):838–855, July 1992.

[14] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, abs/1312.6229, 2013.

[15] T. Serre, L. Wolf, S. M. Bileschi, M. Riesenhuber, and T. Poggio. Robust object recognition with cortex-like mechanisms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(3):411–426, 2007.

[16] F. Song and J. Dongarra. Scaling up matrix computations on shared-memory manycore systems with 1000 cpu cores. In *Proceedings of the 28th ACM International Conference on Supercomputing*, ICS '14, pages 333–342, New York, NY, USA, 2014. ACM.

[17] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton. On the importance of initialization and momentum in deep learning. In *ICML*, volume 28 of *JMLR Proceedings*, pages 1139–1147. JMLR.org, 2013.

[18] C. Szegedy, A. Toshev, and D. Erhan. Deep neural networks for object detection. In C. J. C. Burges, L. Bottou, Z. Ghahramani, and K. Q. Weinberger, editors, *NIPS*, pages 2553–2561, 2013.

[19] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. *CoRR*, abs/1312.4659, 2013.

[20] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders. Segmentation as selective search for object recognition. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 1879–1886, Washington, DC, USA, 2011. IEEE Computer Society.

[21] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *ECCV*, volume 8689 of *Lecture Notes in Computer Science*, pages 818–833. Springer, 2014.