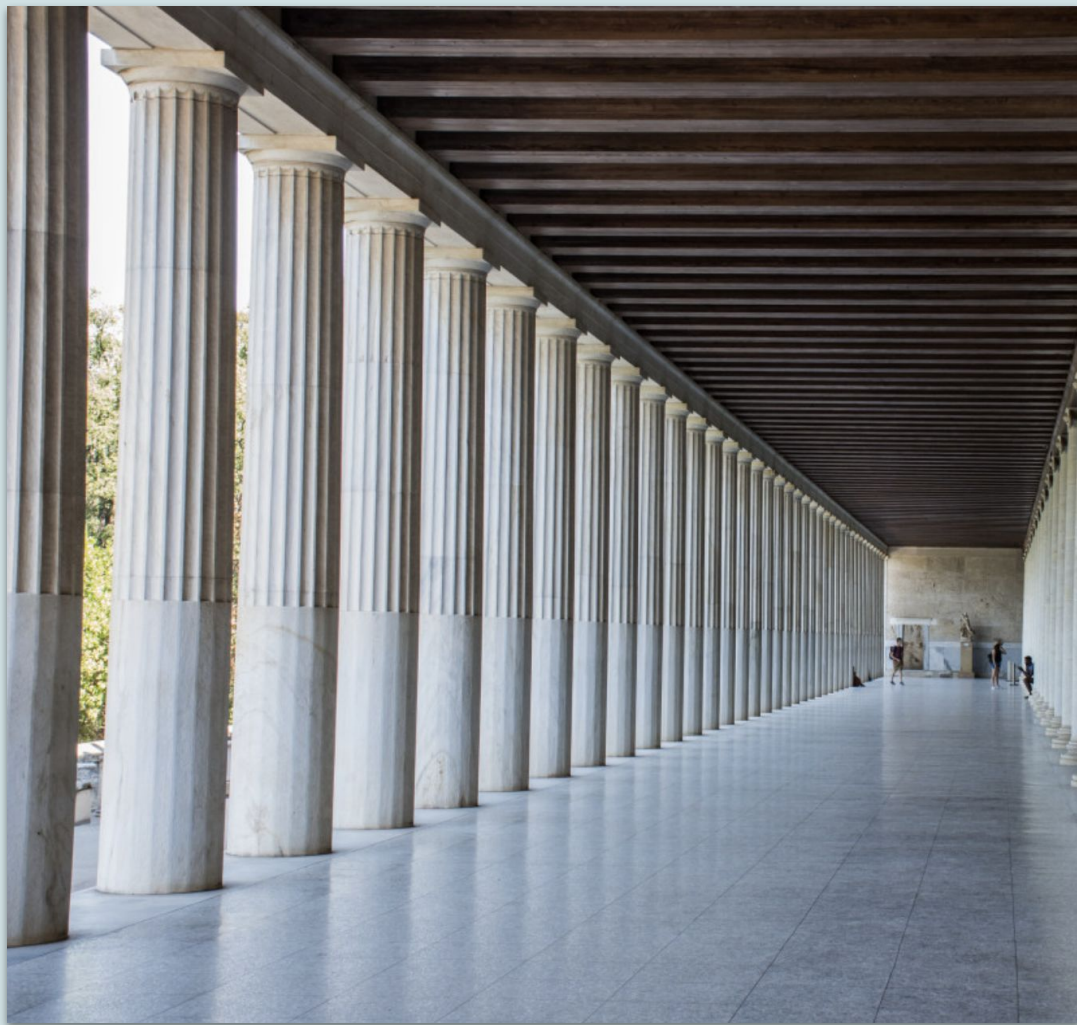
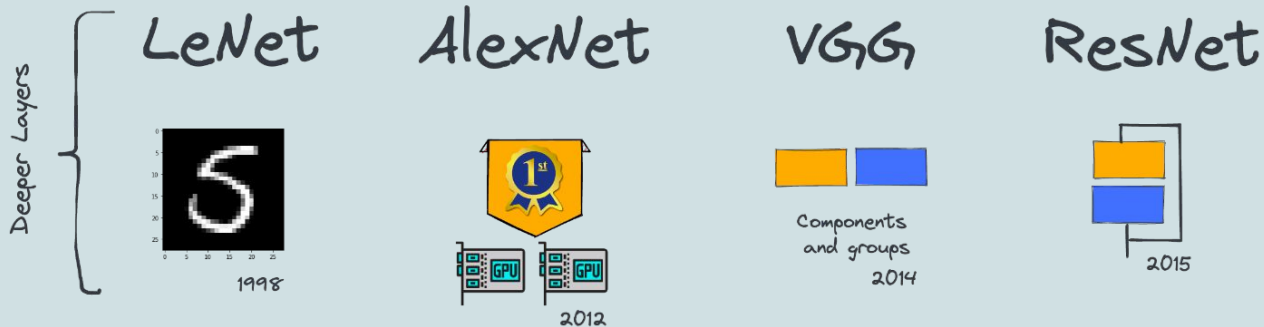


# Going Deeper with CNN II

Study of Classical Architectures

ivanovitch.silva@ufrn.br  
@ivanovitchm





Research understood

- ✓ + accuracy
- ✓ + layers
- ✗ + weight vanishing
- ✗ + weight exploding

## Wide CNN

Inception V1  
2014

New Tools

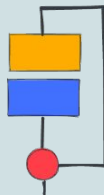
Inception V2, V3  
2015

ResNext  
2015

- ✓ Batch Normalization  
(improve weights vanish, explod.)
- ✓ Dropout  
(attacked the memorization in deeper layers)
- ✗ Overfitting problems persist

## Connectivity Patterns

Residual Block (RB)



DenseNet  
2017

Xception  
2017

SE-Net  
2017

- ✓ reduce memorization without impact to much in complexity
- ✓ Improve RB connectivity or between RB

✗ models without memory restriction

## Mobile CNN

MobileNet v1  
2017

Compact models  
Accuracy vs latency

MobileNet v2  
2018

✓ Compression and Quantization

Shuffle Net  
2017

Refactoring of Convolution

TensorFlow Lite  
2017 >> 2019

✗ Accuracy can be compromised

VGG

#16 #19

arXiv:1409.1556v6 [cs.CV] 10 Apr 2015

## VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan\* & Andrew Zisserman\*

Visual Geometry Group, Department of Engineering Science, University of Oxford  
{karen, az}@robots.ox.ac.uk

### ABSTRACT

In this work we investigate the effect of the convolutional network depth on its accuracy in the large-scale image recognition setting. Our main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small ( $3 \times 3$ ) convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16–19 weight layers. These findings were the basis of our ImageNet Challenge 2014 submission, where our team secured the first and the second places in the localisation and classification tracks respectively. We also show that our representations generalise well to other datasets, where they achieve state-of-the-art results. We have made our two best-performing ConvNet models publicly available to facilitate further research on the use of deep visual representations in computer vision.

### 1 INTRODUCTION

Convolutional networks (ConvNets) have recently enjoyed a great success in large-scale image and video recognition (Krizhevsky et al., 2012; Zeiler & Fergus, 2013; Sermanet et al., 2014; Simonyan & Zisserman, 2014) which has become possible due to the large public image repositories, such as ImageNet (Deng et al., 2009), and high-performance computing systems, such as GPUs or large-scale distributed clusters (Dean et al., 2012). In particular, an important role in the advance of deep visual recognition architectures has been played by the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) (Russakovsky et al., 2014), which has served as a testbed for a few generations of large-scale image classification systems, from high-dimensional shallow feature encodings (Perronnin et al., 2010) (the winner of ILSVRC-2011) to deep ConvNets (Krizhevsky et al., 2012) (the winner of ILSVRC-2012).

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.,





## Karen Simonyan

Co-Founder and Chief Scientist at Inflection AI

Verified email at inflection.ai - [Homepage](#)

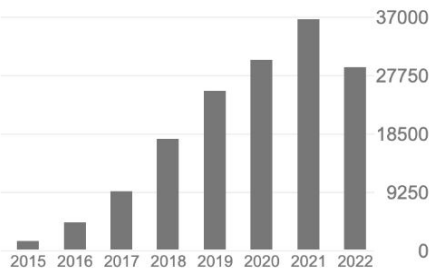
[Artificial Intelligence](#) [Deep Learning](#)



TITLE	CITED BY	YEAR
<a href="#">Very Deep Convolutional Networks for Large-Scale Image Recognition</a> K Simonyan, A Zisserman arXiv preprint arXiv:1409.1556	89068	2014
<a href="#">Mastering the game of go without human knowledge</a> D Silver, J Schrittwieser, K Simonyan, I Antonoglou, A Huang, A Guez, ... nature 550 (7676), 354-359	8082	2017
<a href="#">Two-stream convolutional networks for action recognition in videos</a> K Simonyan, A Zisserman Advances in neural information processing systems 27, 568-576	7288	2014
<a href="#">Spatial Transformer Networks</a> M Jaderberg, K Simonyan, A Zisserman, K Kavukcuoglu arXiv preprint arXiv:1506.02025	6461	2015
<a href="#">Wavenet: A generative model for raw audio</a> A Van Den Oord, S Dieleman, H Zen, K Simonyan, O Vinyals, A Graves, ... arXiv preprint arXiv:1609.03499	6245 *	2016
<a href="#">Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps</a> K Simonyan, A Vedaldi, A Zisserman arXiv preprint arXiv:1312.6034	5800	2013

### Cited by

	All	Since 2017
Citations	156658	148474
h-index	52	50
i10-index	67	64



### Public access

[VIEW ALL](#)

0 articles

7 articles

not available

available

Based on funding mandates



## Andrew Zisserman

University of Oxford

E-mail confirmado em robots.ox.ac.uk - [Página inicial](#)

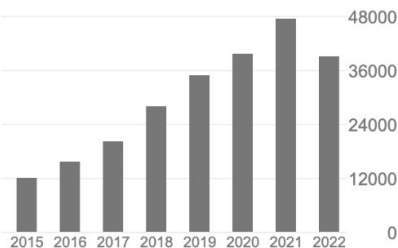
[Computer Vision](#) [Machine Learning](#)



Citado por

[VER TODOS](#)

	Todos	Desde 2017
Citações	323033	209658
Índice h	182	120
Índice i10	580	428



Acesso público

[VER TODOS](#)



Com base nas autorizações de financiamento

Coautores

[VER TODOS](#)

	Karen Simonyan Co-Founder and Chief Scientist ...	>
	Andrea Vedaldi University of Oxford	>

TÍTULO	CITADO POR	ANO
<b>Very deep convolutional networks for large-scale image recognition</b> K Simonyan, A Zisserman arXiv preprint arXiv:1409.1556	88428	2014
<b>Multiple view geometry in computer vision</b> R Hartley, A Zisserman Vision, 2nd ed., New York: Cambridge	31789	2003
<b>The pascal visual object classes (voc) challenge</b> M Everingham, L Van Gool, CKI Williams, J Winn, A Zisserman International journal of computer vision 88 (2), 303-338	15519	2010
<b>Video Google: A text retrieval approach to object matching in videos</b> J Sivic, A Zisserman Computer Vision, 2003. ICCV 2003. IEEE International Conference on, 1470	8123	2003
<b>Two-stream convolutional networks for action recognition in videos</b> K Simonyan, A Zisserman Advances in neural information processing systems 27	7239	2014
<b>Spatial transformer networks</b> M Jaderberg, K Simonyan, A Zisserman Advances in neural information processing systems 28	6420	2015
<b>Deep inside convolutional networks: Visualising image classification models and saliency maps</b> K Simonyan, A Vedaldi, A Zisserman arXiv preprint arXiv:1312.6034	5709	2013
<b>One vladis action recognition? a new model and the kinetics dataset</b>	5544	2017



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 ( $224 \times 224$ RGB image)					
conv3-64	conv3-64 <b>LRN</b>	conv3-64 <b>conv3-64</b>	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 <b>conv3-128</b>	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 <b>conv1-256</b>	conv3-256 conv3-256 <b>conv3-256</b>	conv3-256 conv3-256 conv3-256 <b>conv3-256</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG #16 or  
VGG #19

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

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


- Filter size is constant
- Number of filters increase along the architecture
- Pre-training

```
# import the necessary packages
from tensorflow.keras.applications import VGG16

model = VGG16(weights="imagenet")
```




Conv. layer does not reduce dimensionality




Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0

MaxPool does not reduce depth



block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000

How to solve the parameter explosion problem?



Total params: 138,357,544  
Trainable params: 138,357,544  
Non-trainable params: 0



```
# loop over the layers in the network and display them to the console
for (i, layer) in enumerate(model.layers):
    print("[INFO] {} \t {}".format(i, layer.__class__.__name__))
```

```
[INFO] 0      InputLayer
[INFO] 1      Conv2D
[INFO] 2      Conv2D
[INFO] 3      MaxPooling2D
[INFO] 4      Conv2D
[INFO] 5      Conv2D
[INFO] 6      MaxPooling2D
[INFO] 7      Conv2D
[INFO] 8      Conv2D
[INFO] 9      Conv2D
[INFO] 10     MaxPooling2D
[INFO] 11     Conv2D
[INFO] 12     Conv2D
[INFO] 13     Conv2D
[INFO] 14     MaxPooling2D
[INFO] 15     Conv2D
[INFO] 16     Conv2D
[INFO] 17     Conv2D
[INFO] 18     MaxPooling2D
[INFO] 19     Flatten
[INFO] 20     Dense
[INFO] 21     Dense
[INFO] 22     Dense
```







# How to Use 1x1 Convolutions to Manage Model Complexity

312.4400v3 [cs.NE] 4 Mar 2014

---

## Network In Network

---

Min Lin<sup>1,2</sup>, Qiang Chen<sup>2</sup>, Shuicheng Yan<sup>2</sup>

<sup>1</sup>Graduate School for Integrative Sciences and Engineering

<sup>2</sup>Department of Electronic & Computer Engineering

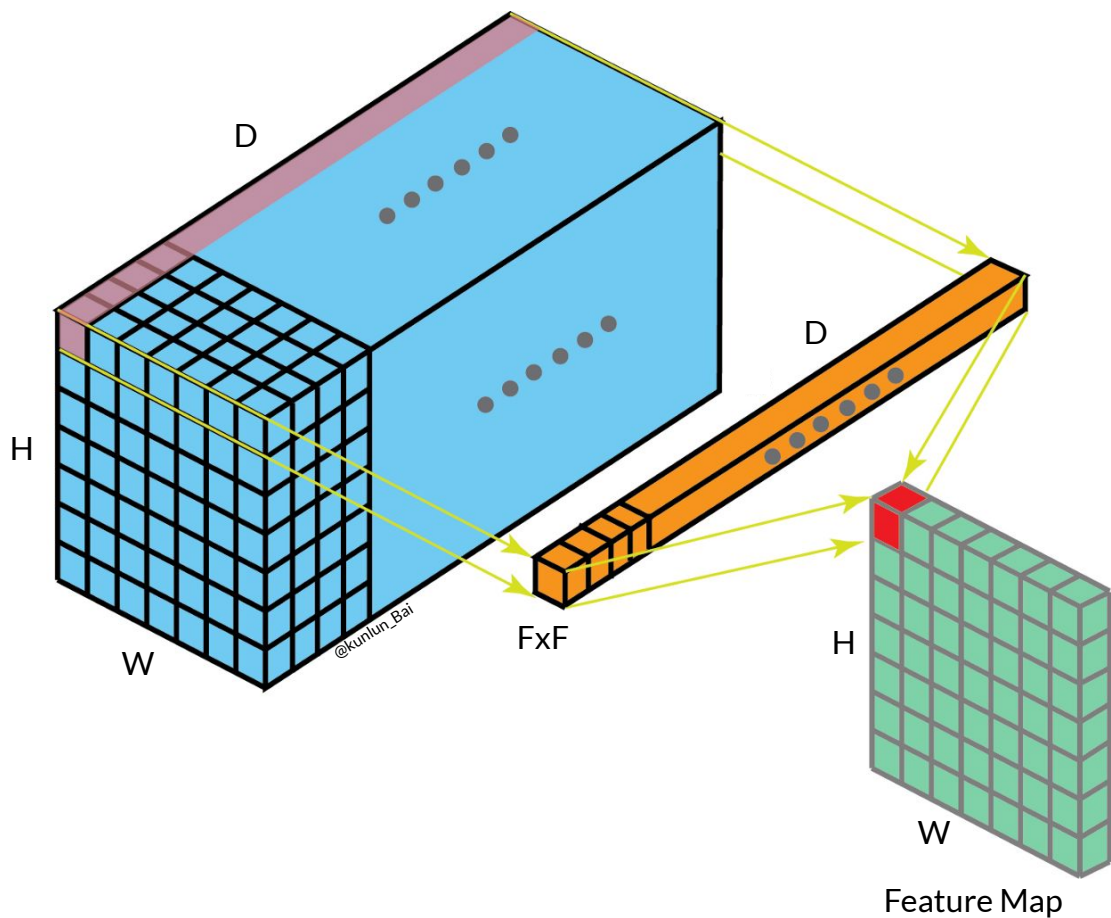
National University of Singapore, Singapore

{linmin, chenqiang, eleyans}@nus.edu.sg

### Abstract

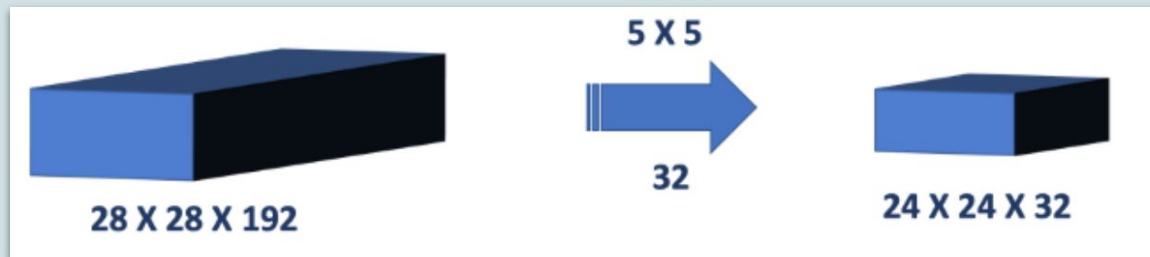
We propose a novel deep network structure called “Network In Network”(NIN) to enhance model discriminability for local patches within the receptive field. The conventional convolutional layer uses linear filters followed by a nonlinear activation function to scan the input. Instead, we build micro neural networks with more complex structures to abstract the data within the receptive field. We instantiate the micro neural network with a multilayer perceptron, which is a potent function approximator. The feature maps are obtained by sliding the micro networks over the input in a similar manner as CNN; they are then fed into the next layer. Deep NIN can be implemented by stacking multiple of the above described structure. With enhanced local modeling via the micro network, we are able to utilize global average pooling over feature maps in the classification layer, which is easier to interpret and less prone to overfitting than traditional fully connected layers. We demonstrated the state-of-the-art classification performances with NIN on CIFAR-10 and CIFAR-100, and reasonable performances on SVHN and MNIST datasets.

### 1 Introduction



## 1x1 convolutional layer

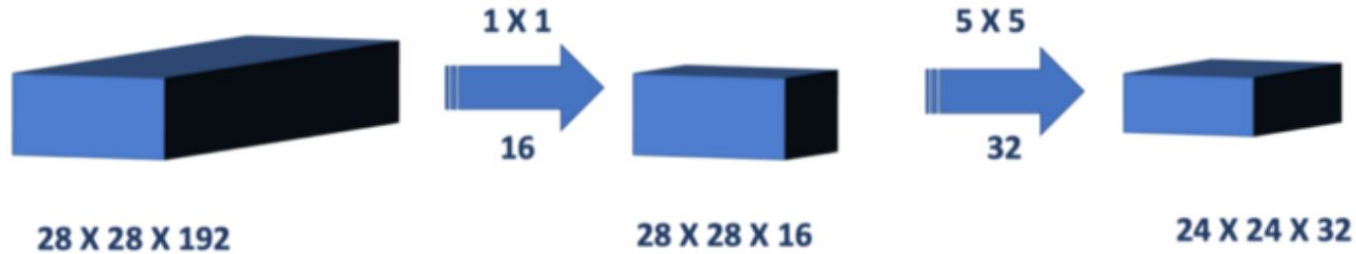
The depth of the output of one convolutional layer is only defined by the number of parallel filters applied to the input.



```
# example of a 1x1 filter for dimensionality reduction
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D
# create model
model = Sequential()
model.add(Conv2D(32, (5,5), padding="valid", activation="relu", input_shape=(28, 28, 192)))
# summarize model
model.summary()
```

Layer (type)	Output Shape	Param #
conv2d_17 (Conv2D)	(None, 24, 24, 32)	153632
Total params: 153,632		
Trainable params: 153,632		
Non-trainable params: 0		





```
# create model
model = Sequential()
model.add(Conv2D(16, (1,1), activation="relu", input_shape=(28, 28, 192)))
model.add(Conv2D(32, (5,5), padding="valid", activation="relu"))
# summarize model
model.summary()
```

Layer (type)	Output Shape	Param #
=====		
conv2d_14 (Conv2D)	(None, 28, 28, 16)	3088
-----		
conv2d_15 (Conv2D)	(None, 24, 24, 32)	12832
=====		
Total params: 15,920		
Trainable params: 15,920		
Non-trainable params: 0		



---

## Going deeper with convolutions

---

**Christian Szegedy**

Google Inc.

**Wei Liu**

University of North Carolina, Chapel Hill

**Yangqing Jia**

Google Inc.

**Pierre Sermanet**

Google Inc.

**Scott Reed**

University of Michigan

**Dragomir Anguelov**

Google Inc.

**Dumitru Erhan**

Google Inc.

**Vincent Vanhoucke**

Google Inc.

**Andrew Rabinovich**

Google Inc.

### Abstract

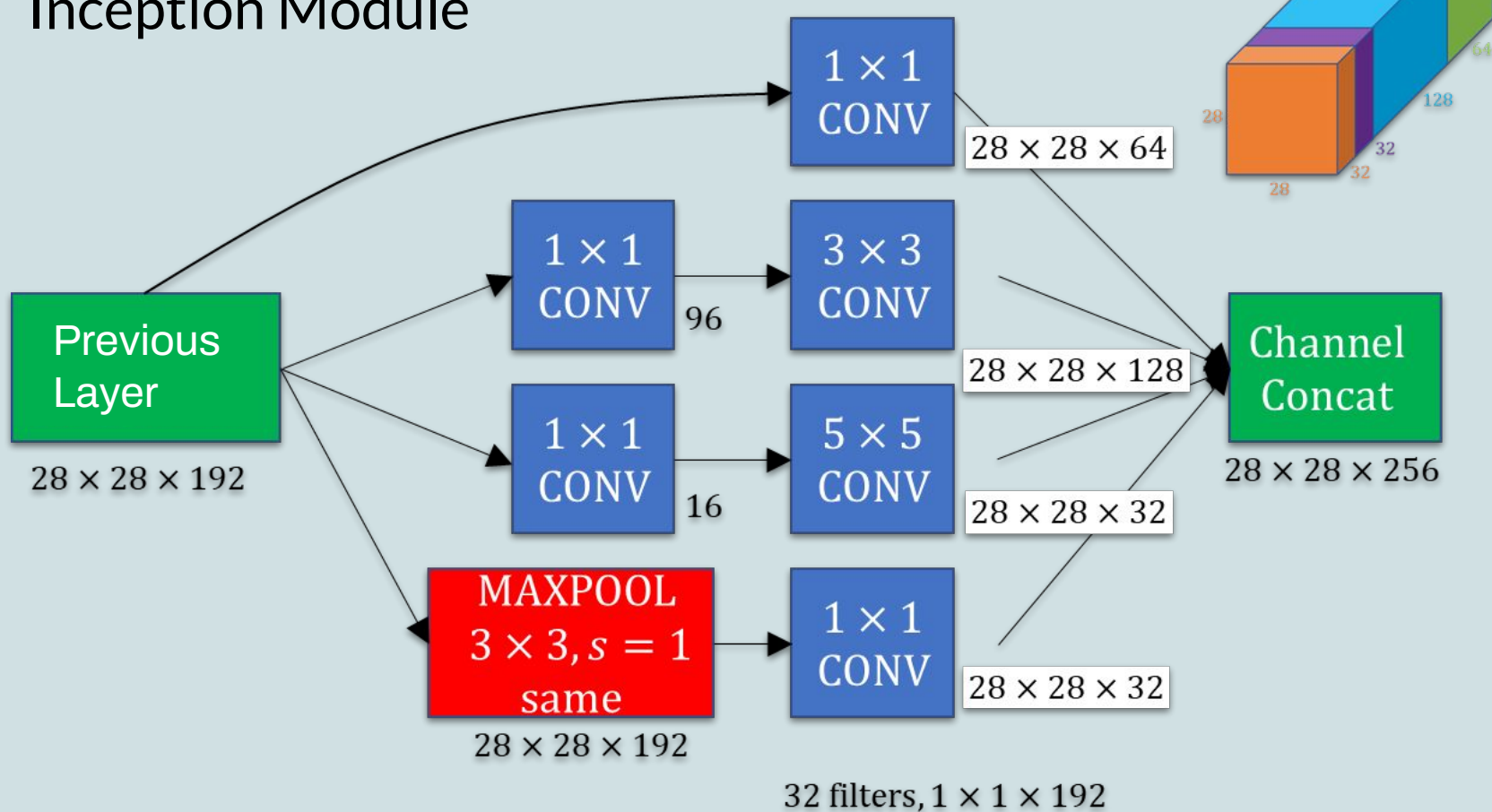
We propose a deep convolutional neural network architecture codenamed Inception, which was responsible for setting the new state of the art for classification and detection in the ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC14). The main hallmark of this architecture is the improved utilization of the computing resources inside the network. This was achieved by a carefully crafted design that allows for increasing 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, mainly due to the advances of deep learning, more concretely convolutional networks [10], the quality of image recognition and object detection has been progressing at a dramatic pace. 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

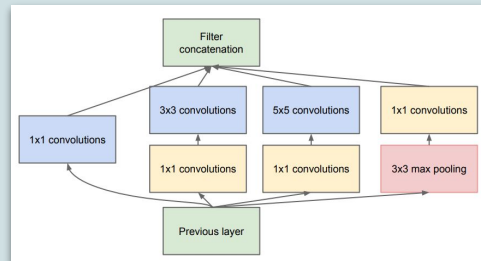


# Inception Module



# GoogLeNet incarnation of the Inception architecture

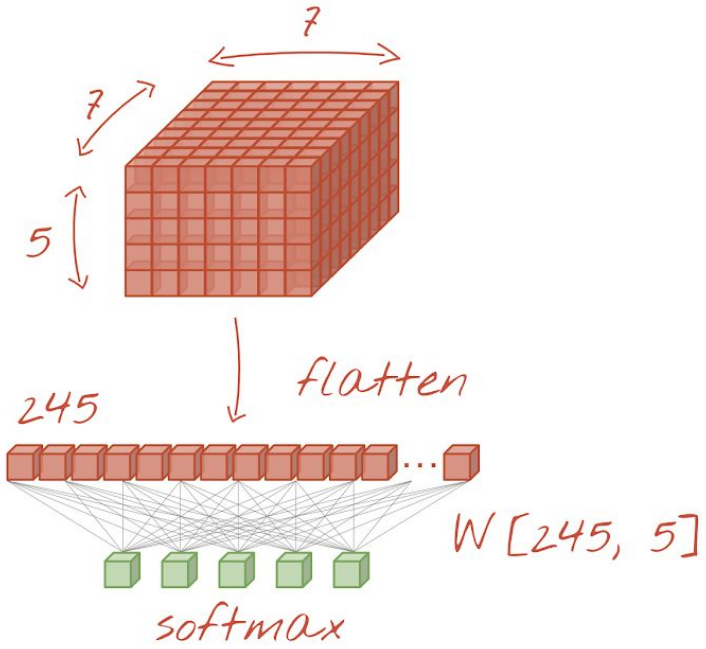
type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	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								



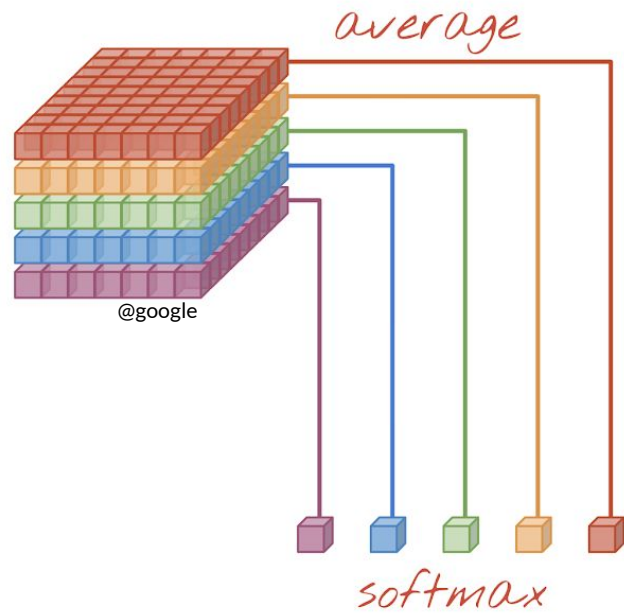
+ - 6.8M



Fully connected layer



Global average pooling



1225 weights *cheaper* → 0 weights