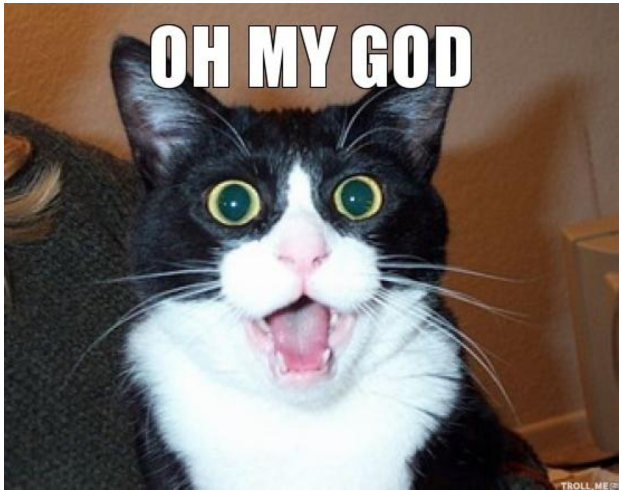
Aula 6:
Redes Convolucionais – Parte II - VGG, GoogLeNet, Inception Module, ResNet, Highways networks……………….

www.deeplearningbrasil.com.br



Relembrando…

Desafio ImageNet: top-5 error

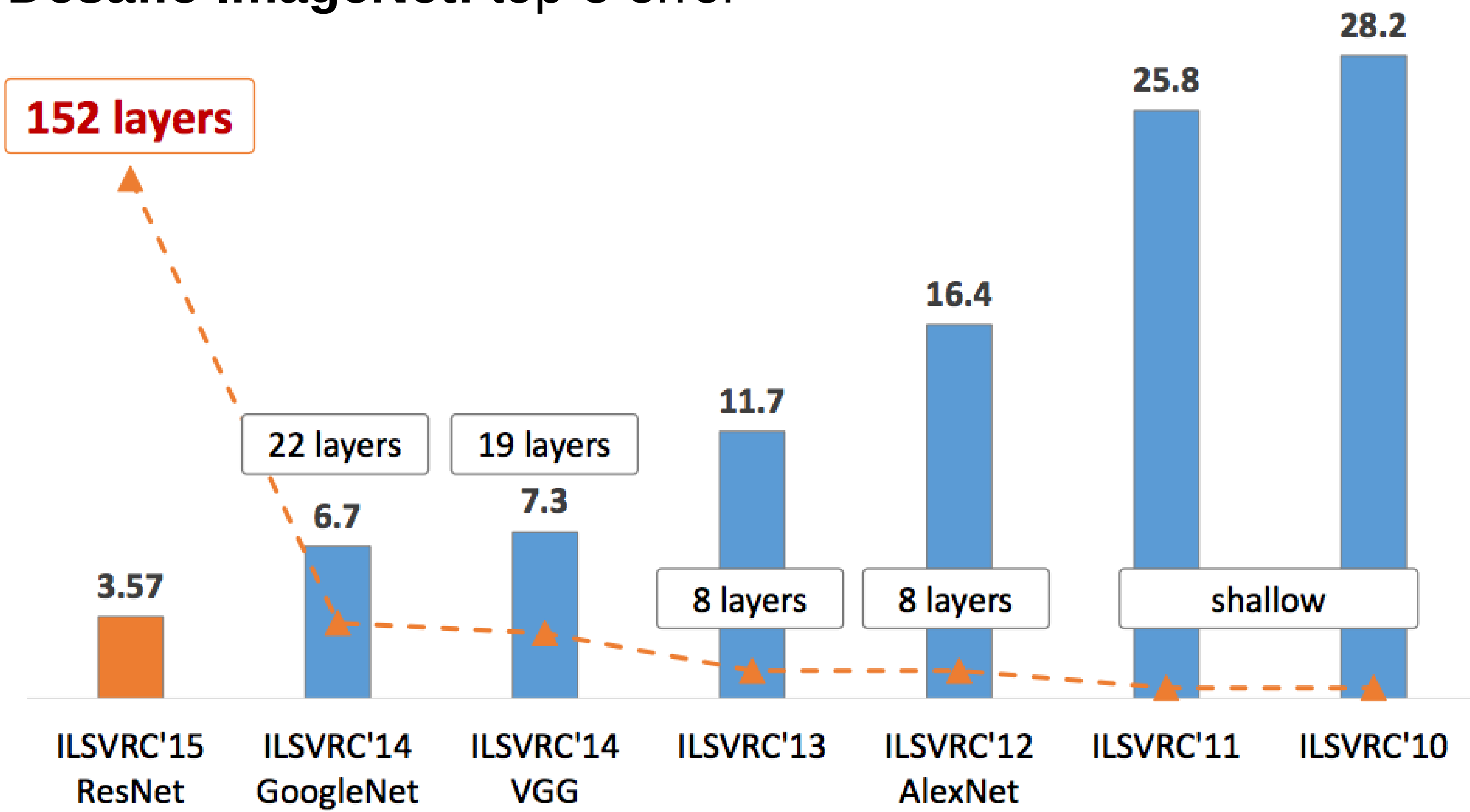
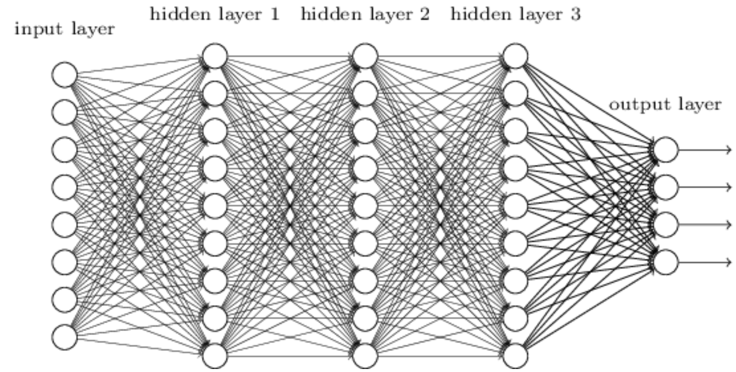
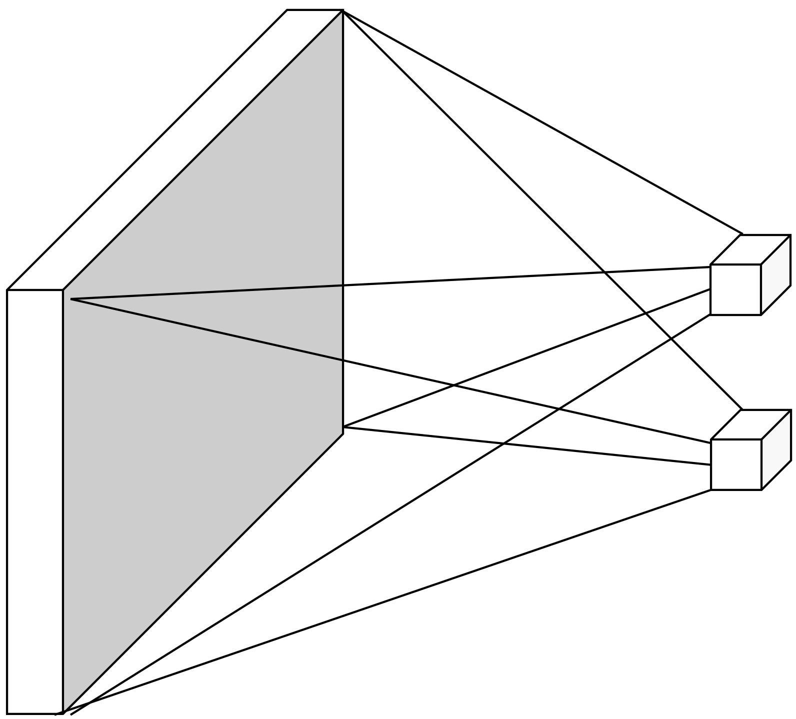


Figure source: [Kaiming He]()

Arquitetura antes do conceito de Deep Learning

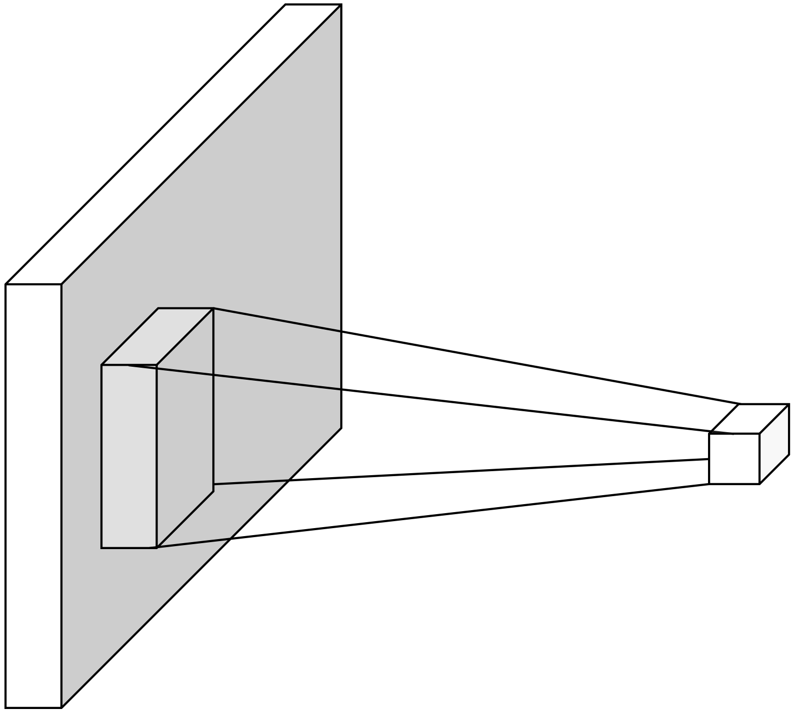




image

Fully connected layer

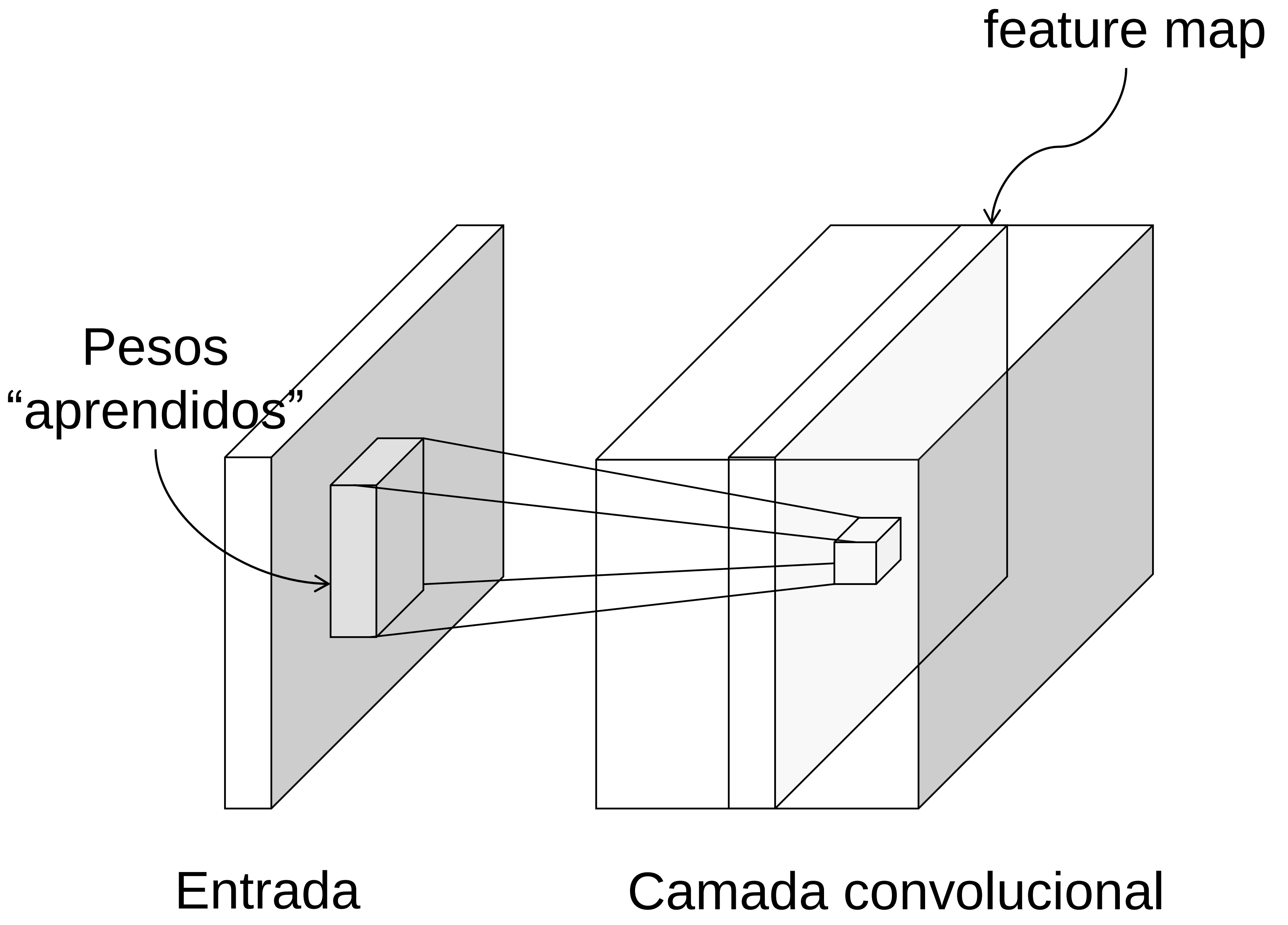
Arquitetura convolucional



image

Convolutional layer

Arquitetura convolucional



Arquitetura convolucional da AlexNet

- Relembrando as camadas:
  - Entrada 224x224x3
  - Primeiro filtro 11x11x3
  - Filtros convolucionais menores a medida que aumenta a profundidade
  - Ex.: 5x5, 3x3

  - O que acontece se usarmos apenas filtros pequenos? (3x3, por exemplo?)

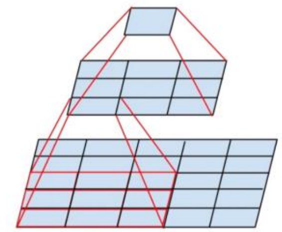
VGGNet: ILSVRC 2014 2^{nd} Lugar

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

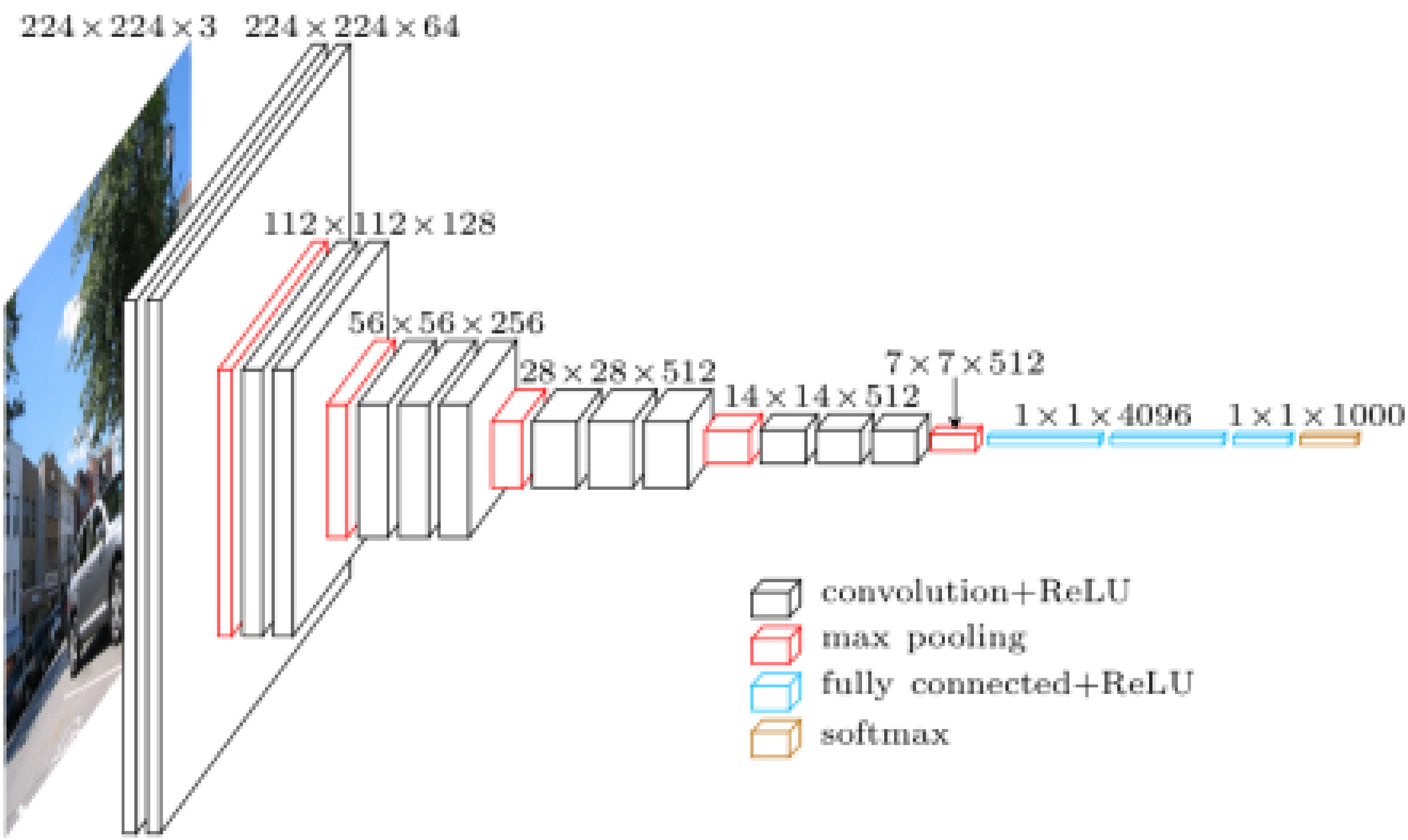
- Sequence of deeper networks trained progressively
- Campos receptivos substituídos por filtros 3x3 (com ReLU entre eles)



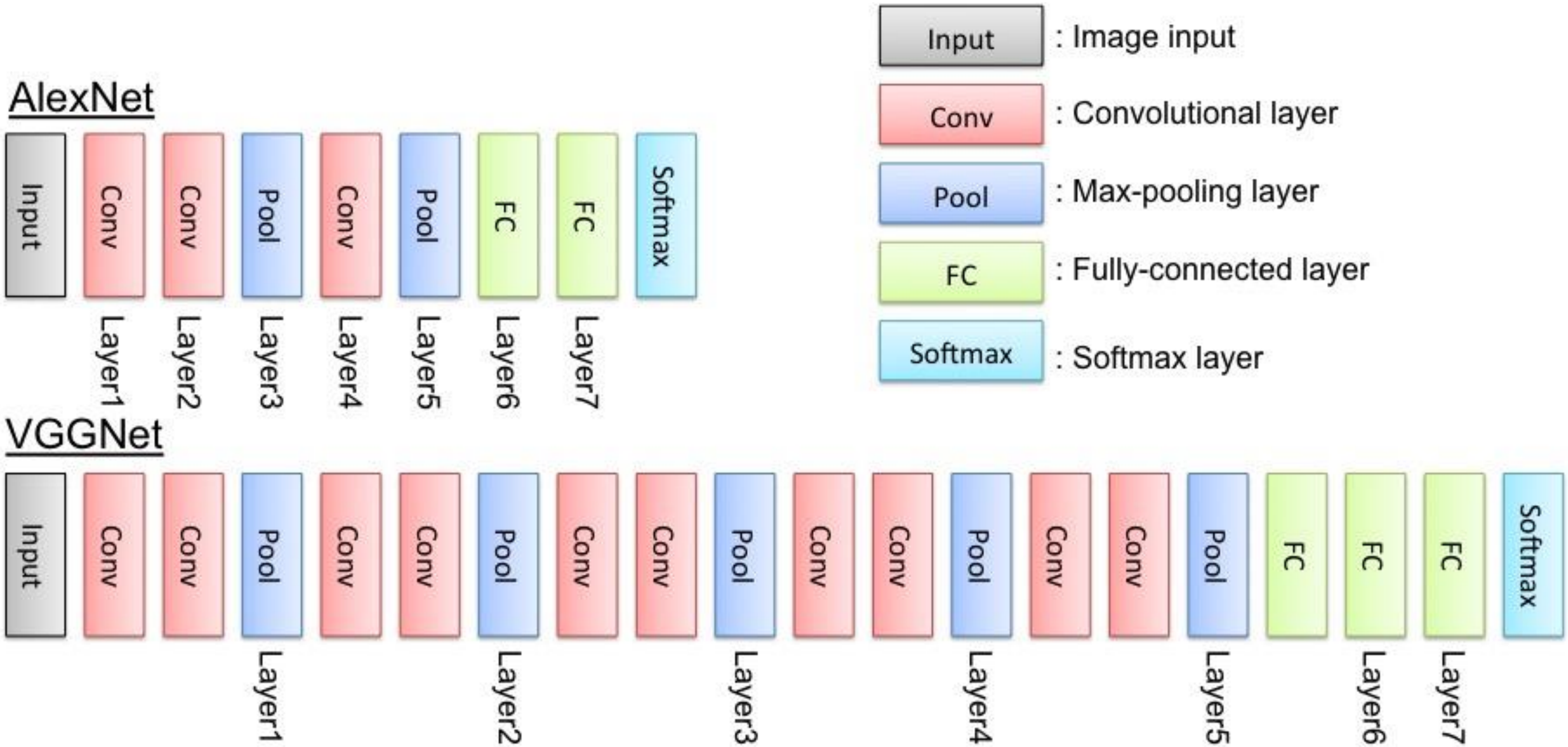
- Uma camada 7x7 com C mapas precisa de $49C^2$ pesos, camadas 3x3 precisam de $27C^2$ pesos

K. Simonyan and A. Zisserman, [Very Deep Convolutional Networks for Large-Scale Image Recognition], ICLR 2015

VGGNet: ILSVRC 2014 2nd Lugar



Características da VGGNet



Codificando...

ImageNet classification with Python and Keras

```python
# import the necessary packages
from keras.preprocessing import image as image_utils
from imagenet_utils import decode_predictions
from imagenet_utils import preprocess_input
from vgg16 import VGG16
import numpy as np
import argparse
import cv2

# construct the argument parse and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", required=True,
    help="path to the input image")
args = vars(ap.parse_args())

# load the original image via OpenCV so we can draw on it and display
# it to our screen later
orig = cv2.imread(args["image"])
```

Exemplo obtido de:
http://www.pyimagesearch.com/2016/08/10/imagenet-classification-with-python-and-keras/

Codificando...

ImageNet classification with Python and Keras — Python

```python
# load the input image using the Keras helper utility while ensuring
# that the image is resized to 224x224 pxiels, the required input
# dimensions for the network -- then convert the PIL image to a
# NumPy array
print("[INFO] loading and preprocessing image...")
image = image_utils.load_img(args["image"], target_size=(224, 224))
image = image_utils.img_to_array(image)
```

Codificando...

ImageNet classification with Python and Keras

Python

```python
# our image is now represented by a NumPy array of shape (3, 224, 224),
# but we need to expand the dimensions to be (1, 3, 224, 224) so we can
# pass it through the network -- we'll also preprocess the image by
# subtracting the mean RGB pixel intensity from the ImageNet dataset
image = np.expand_dims(image, axis=0)
image = preprocess_input(image)
```

Codificando...

VGG16, VGG19 ou ResNet50

ImageNet classification with Python and Keras — Python

```python
# load the VGG16 network
print("[INFO] loading network...")
model = VGG16(weights="imagenet")

# classify the image
print("[INFO] classifying image...")
preds = model.predict(image)
(inID, label) = decode_predictions(preds)[0]

# display the predictions to our screen
print("ImageNet ID: {}, Label: {}".format(inID, label))
cv2.putText(orig, "Label: {}".format(label), (10, 30),
    cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2)
cv2.imshow("Classification", orig)
cv2.waitKey(0)
```

$ python test_imagenet.py --image images/dog_beagle.png

Caramba...
esse curso é demais...
Não preciso de mais nada...

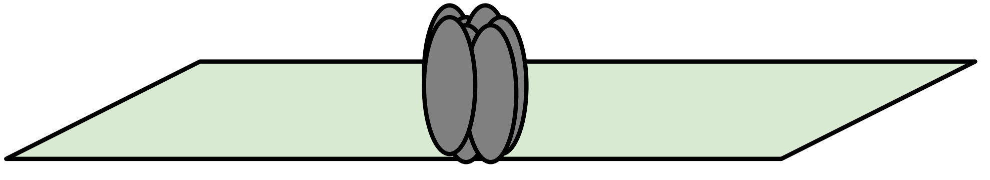
Calma...

GoogLeNet: ILSVRC 2014 winner

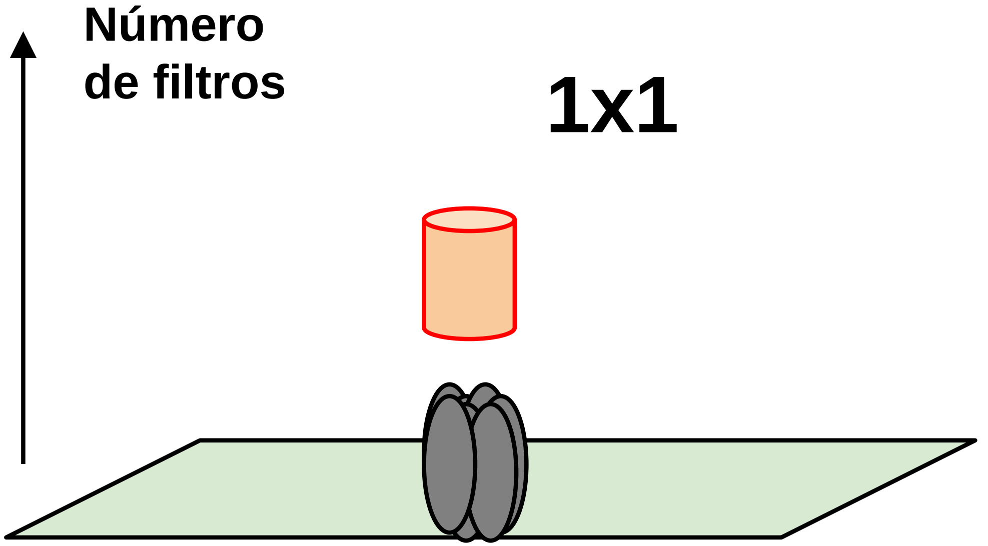
http://knowyourmeme.com/memes/we-need-to-go-deeper

C. Szegedy et al., Going deeper with convolutions, CVPR 2015

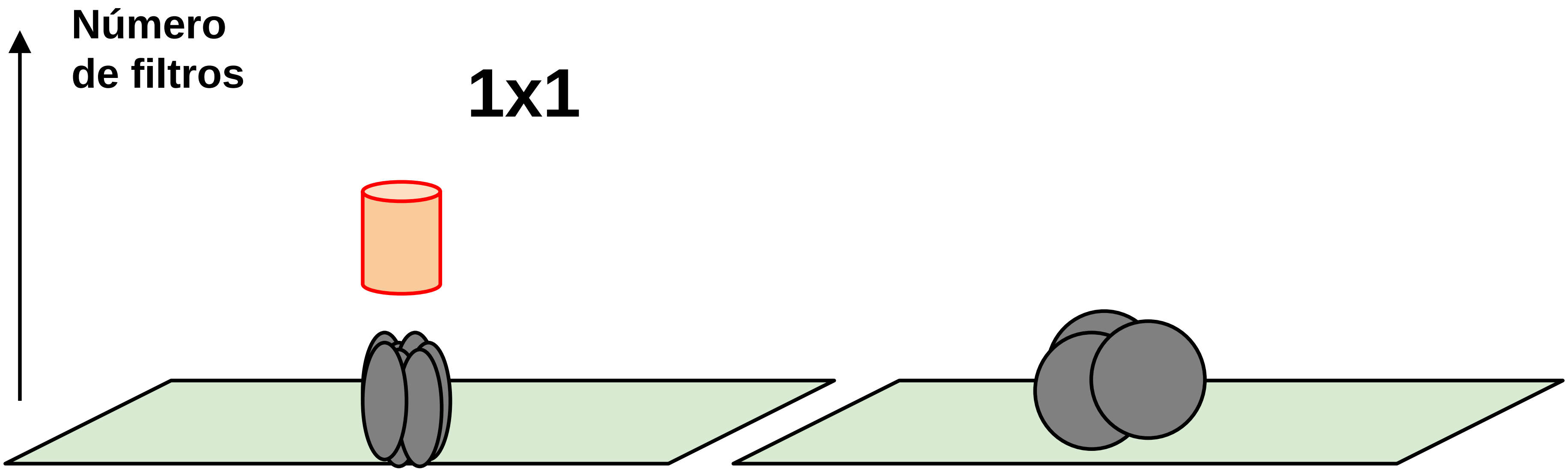
Em imagens, correlações tender a ser local



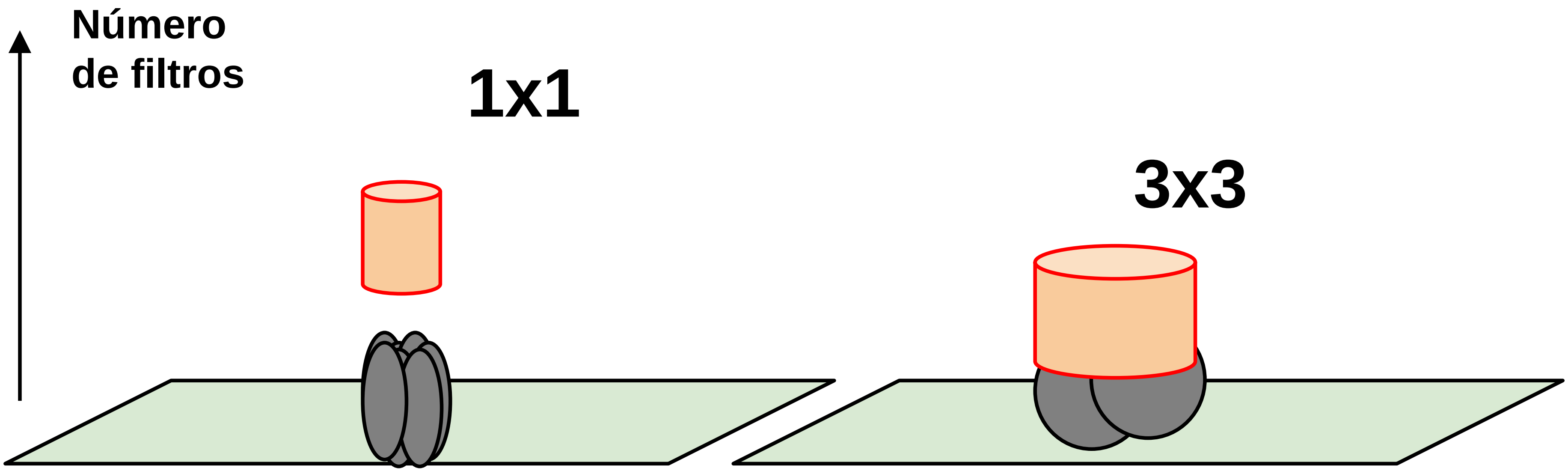
Ideia: cobrir clusters locais com convoluções 1x1



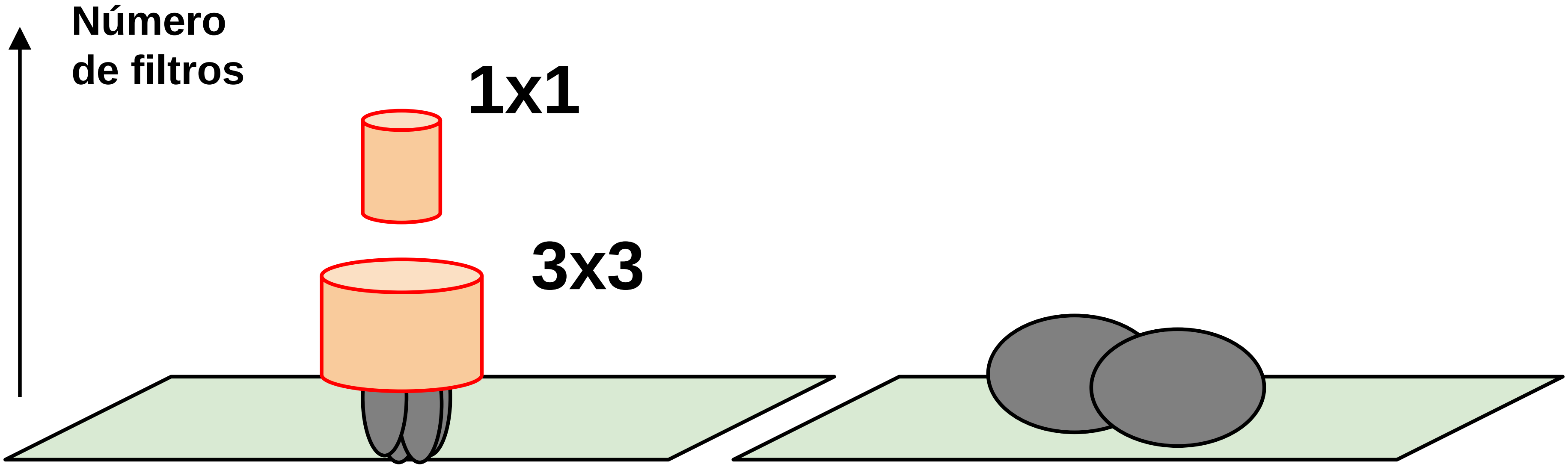
Correlações menos espalhadas



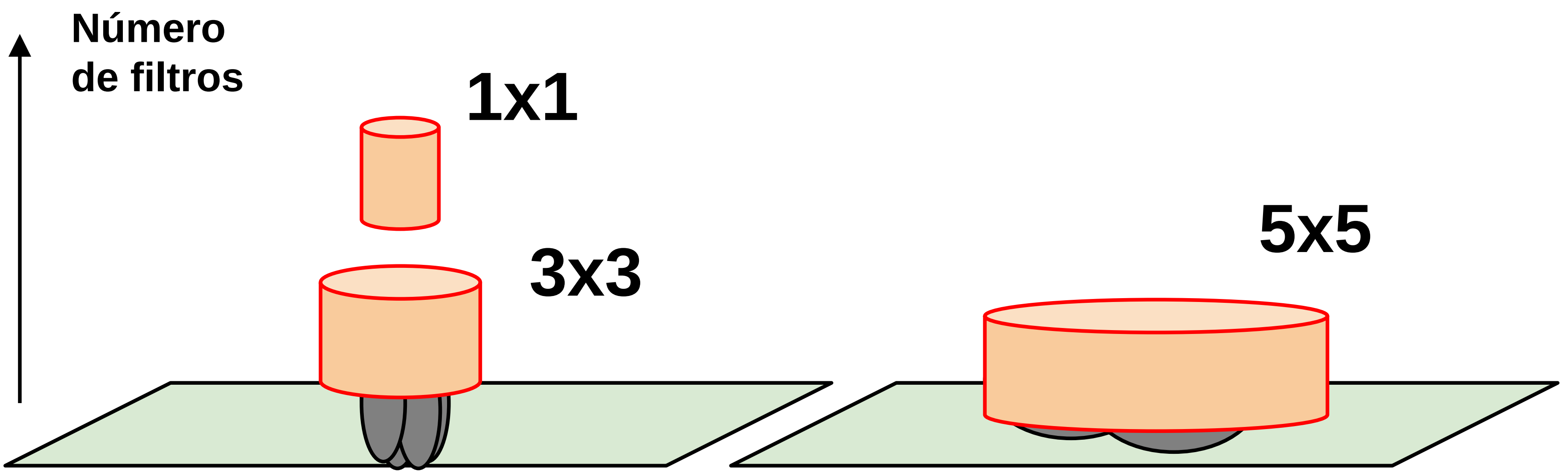
Correlações menos espalhadas



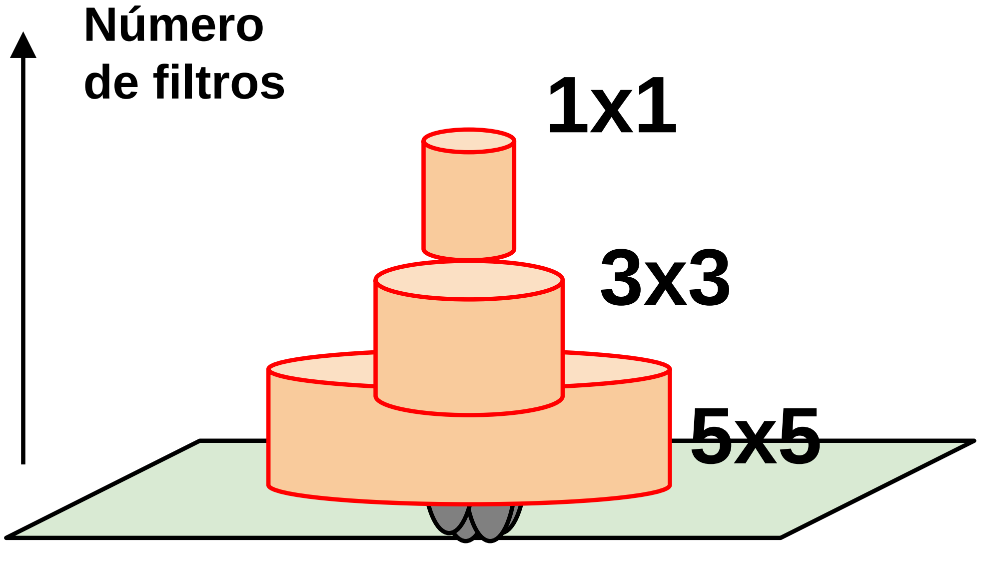
Correlações menos espalhadas



Correlações menos espalhadas

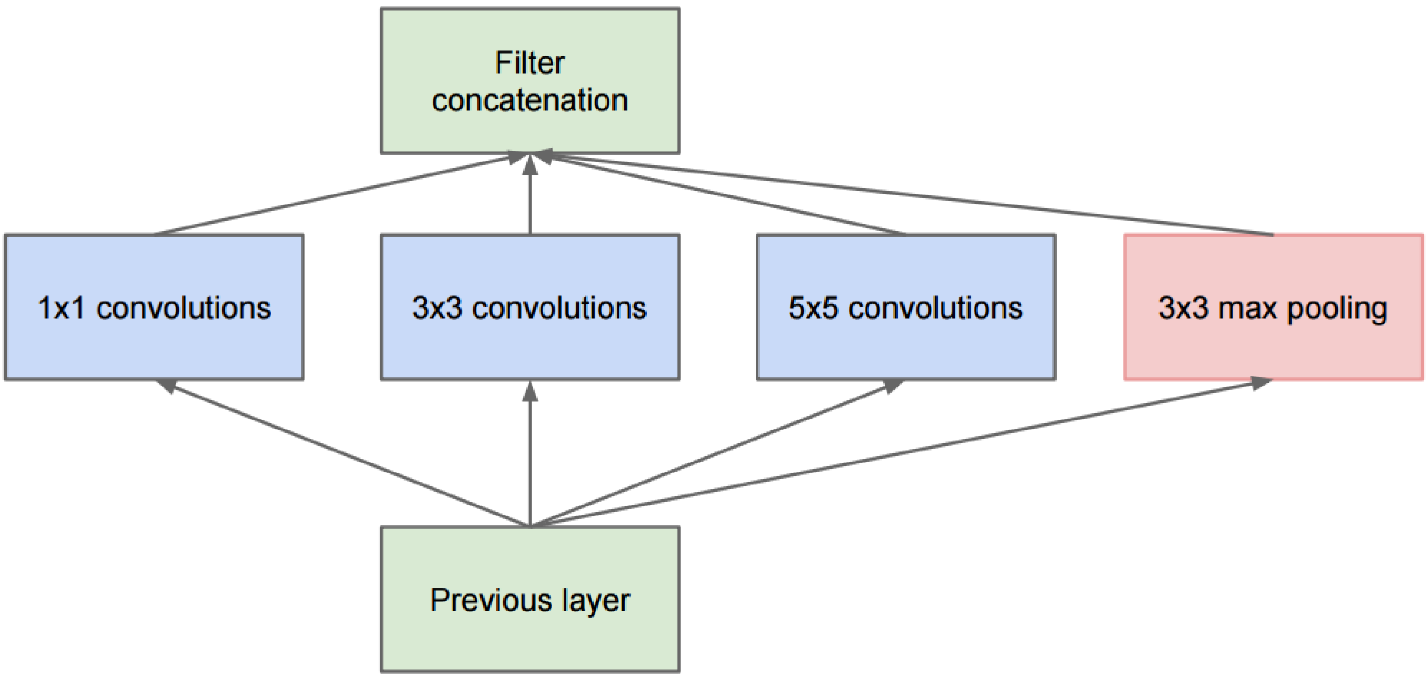


SLIDE CREDIT:GOOGLE INC


Número
de filtros
1x1
3x3
5x5

GoogLeNet

- Inception Module v1
  - Filtros convolucionais em paralelo com campos receptivos diferentes
  - Usa filtros 1x1 para redução de dimensionalidade antes de operações de “alto custo” antes das convoluções maiores



GoogLeNet

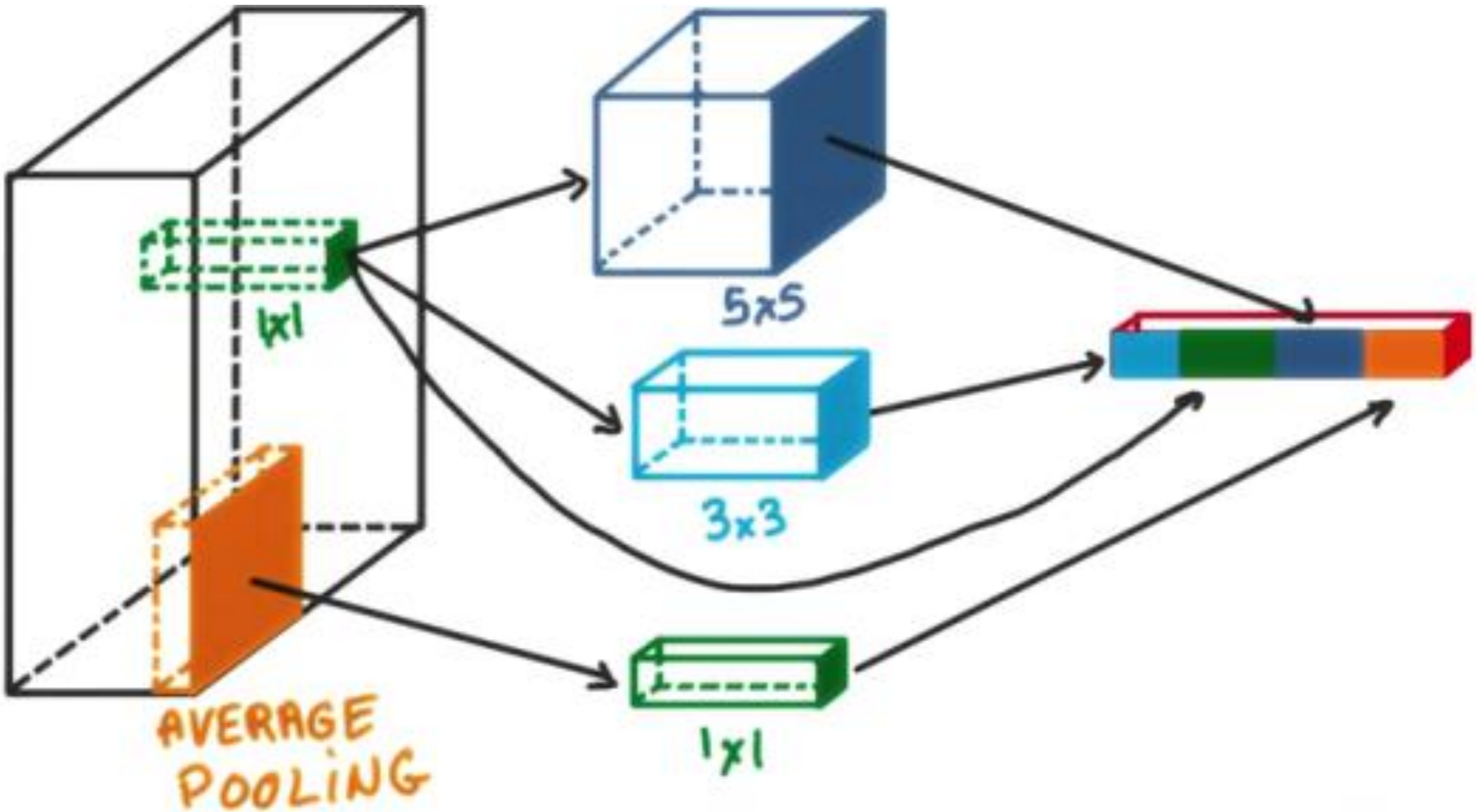
Convolução 1x1

- Reduz o esforço computacional antes de filtros 3x3 e 5x5
- Inclui o uso de RELU

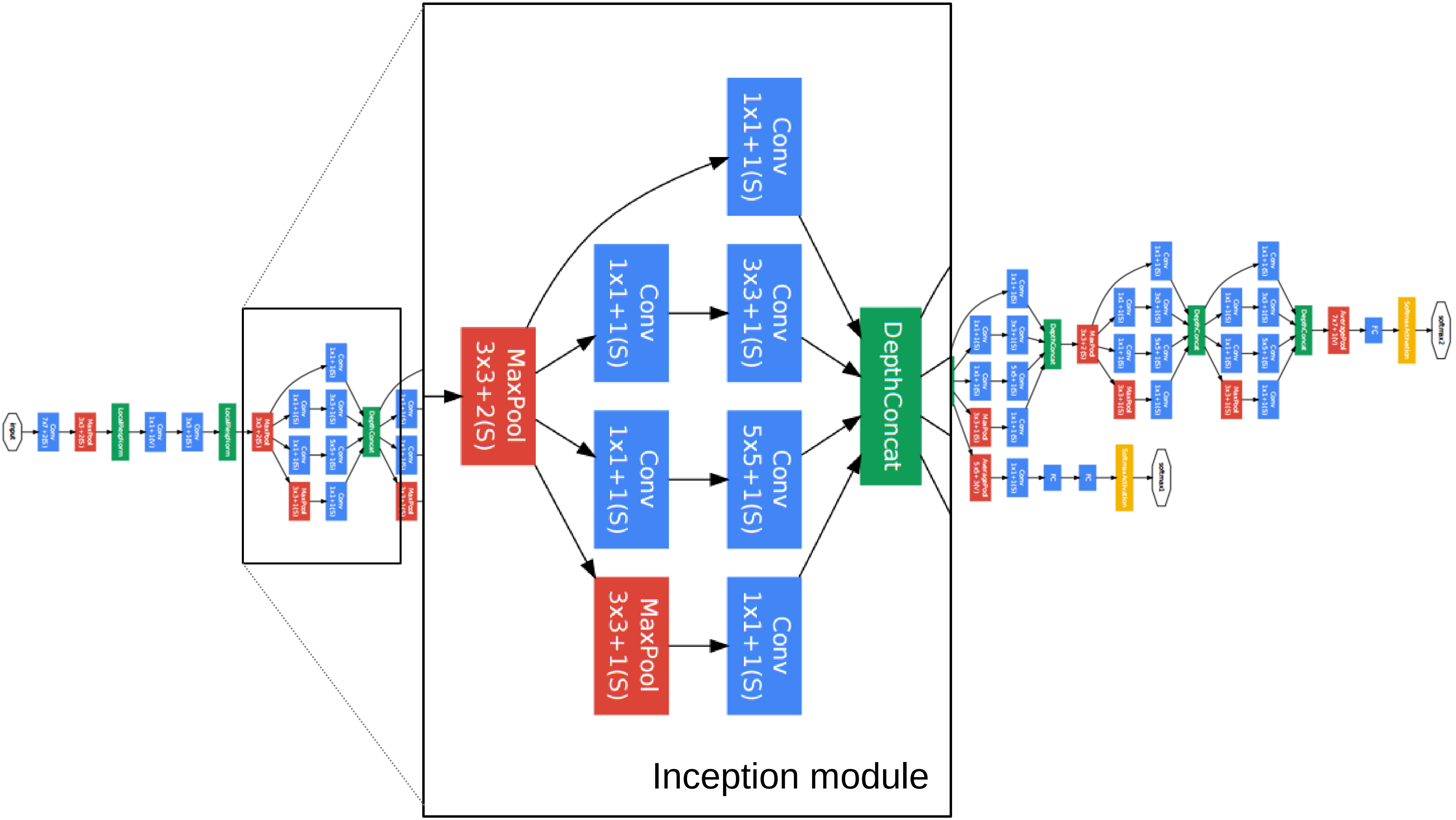
- Exemplo:
- Entrada 256 mapas de filtros 56x56
- Inserção de camada 1x1x64
- 56x56x256 -> 56x56x64 (redução de 4x)

http://iamaaditya.github.io/2016/03/one-by-one-convolution/

GoogLeNet

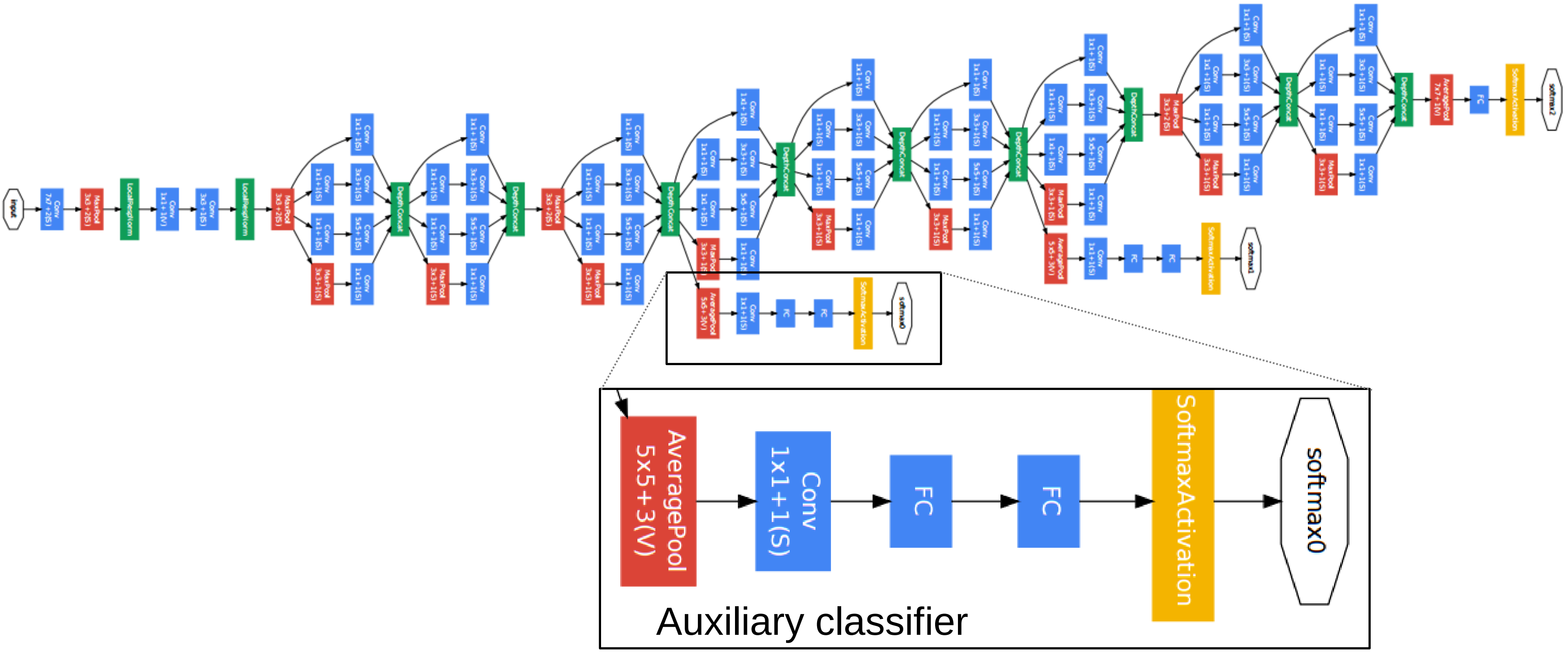


GoogLeNet



C. Szegedy et al., [Going deeper with convolutions](), CVPR 2015

GoogLeNet



C. Szegedy et al., [Going deeper with convolutions](), CVPR 2015

GoogLeNet

An alternative view:

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								

C. Szegedy et al., Going deeper with convolutions, CVPR 2015

Computer Science > Computer Vision and Pattern Recognition

Rethinking the Inception Architecture for Computer Vision

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, Zbigniew Wojna

(Submitted on 2 Dec 2015 (v1), last revised 11 Dec 2015 (this version, v3))

Convolutional networks are at the core of most state-of-the-art computer vision solutions for a wide variety of tasks. Since 2014 very deep convolutional networks started to become mainstream, yielding substantial gains in various benchmarks. Although increased model size and computational cost tend to translate to immediate quality gains for most tasks (as long as enough labeled data is provided for training), computational efficiency and low parameter count are still enabling factors for various use cases such as mobile vision and big-data scenarios. Here we explore ways to scale up networks in ways that aim at utilizing the added computation as efficiently as possible by suitably factorized convolutions and aggressive regularization. We benchmark our methods on the ILSVRC 2012 classification challenge validation set demonstrate substantial gains over the state of the art: 21.2% top-1 and 5.6% top-5 error for single frame evaluation using a network with a computational cost of 5 billion multiply-adds per inference and with using less than 25 million parameters. With an ensemble of 4 models and multi-crop evaluation, we report 3.5% top-5 error on the validation set (3.6% error on the test set) and 17.3% top-1 error on the validation set.

Princípios de design:

1) Avoid Representational Bottlenecks, Especially Early in the Network
2) Representações dimensionais mais altas são mais fáceis de processar localmente
3) A agregação espacial pode ser feita em dimensões mais baixas sem perda

   Ex.: Antes de 3x3, redução de dimensionalidade
4) Equilíbrio entre profundidade e largura

Ponto chave: Fatorização das convoluções

Inception v2

Inception v2 substituiu as convoluções 5x5 por dois filtros 3x3 sucessivos:

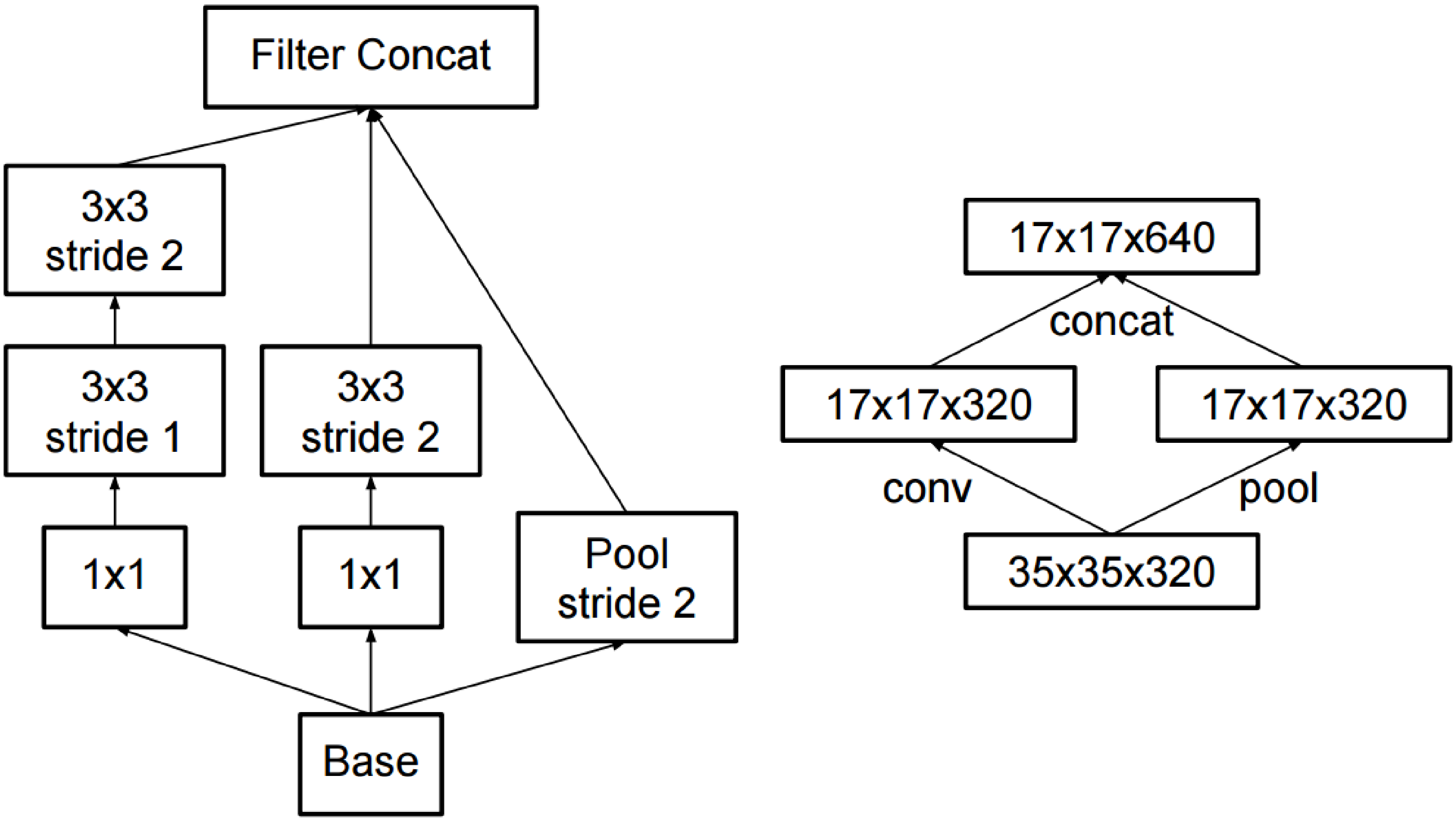


Figura 5

Inception v2

- Mais variantes dos módulos com fatorização mais agressiva de filtros

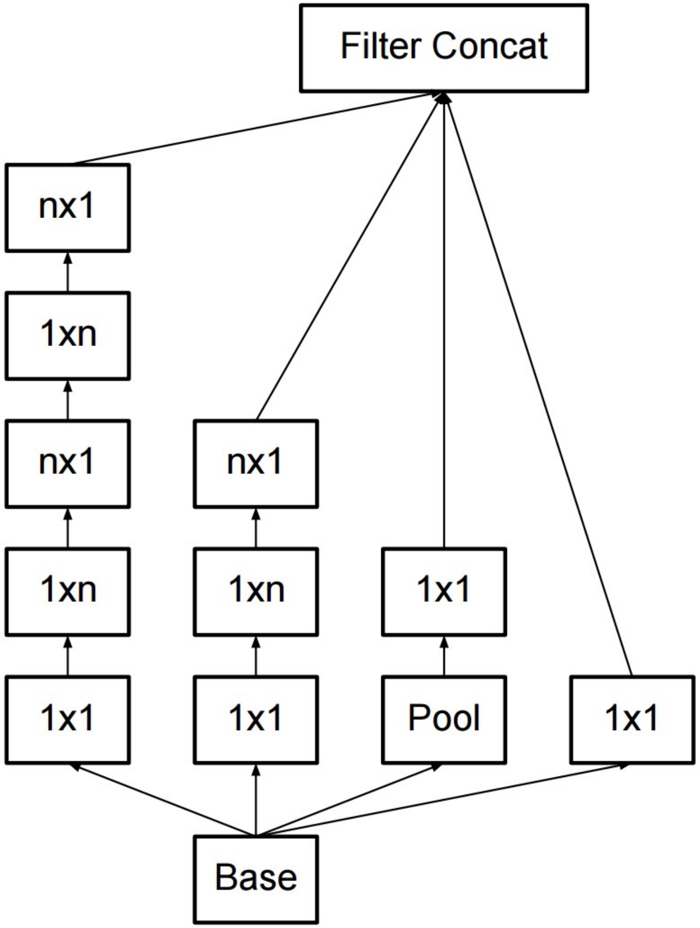


Figura 6

C. Szegedy et al., [Rethinking the inception architecture for computer vision](), CVPR 2016

Inception v2

- Mais variantes dos módulos com fatorização mais agressiva de filtros

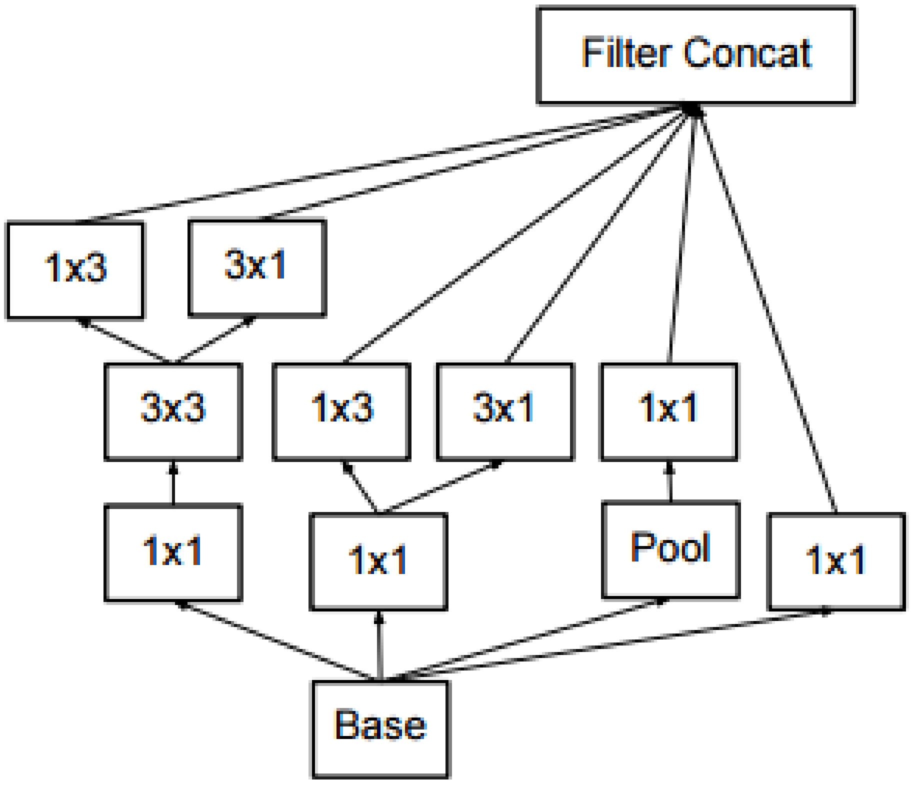


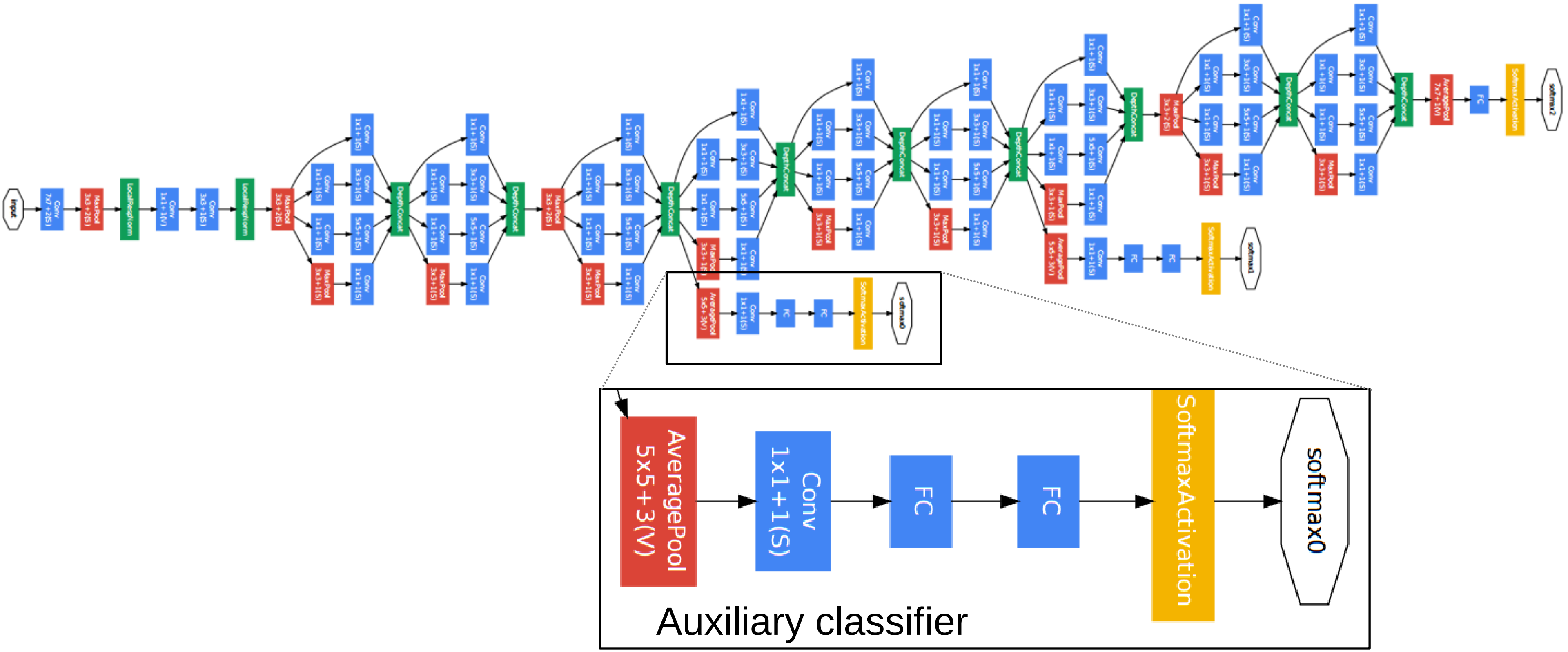
Figura 7

C. Szegedy et al., [Rethinking the inception architecture for computer vision](), CVPR 2016

Inception v2/v3

type	patch size/stride or remarks	input size
conv	3×3/2	299×299×3
conv	3×3/1	149×149×32
conv padded	3×3/1	147×147×32
pool	3×3/2	147×147×64
conv	3×3/1	73×73×64
conv	3×3/2	71×71×80
conv	3×3/1	35×35×192
3×Inception	As in figure 5	35×35×288
5×Inception	As in figure 6	17×17×768
2×Inception	As in figure 7	8×8×1280
pool	8 × 8	8 × 8 × 2048
linear	logits	1 × 1 × 2048
softmax	classifier	1 × 1 × 1000

GoogLeNet – Inception-V3

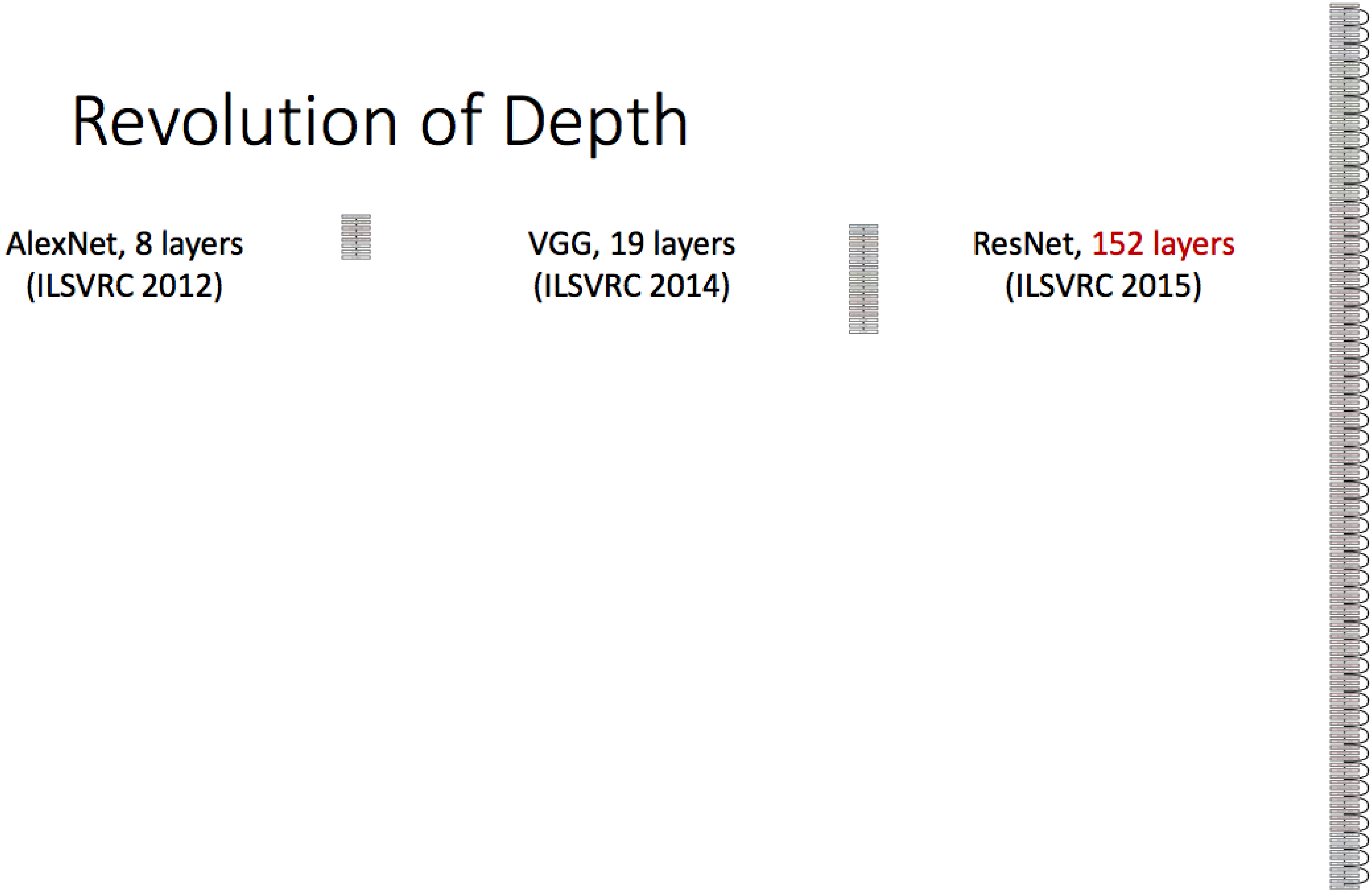


C. Szegedy et al., [Going deeper with convolutions](), CVPR 2015


I WAS WINNING
IMAGENET
UNTIL A
DEEPER MODEL
CAME ALONG
imgflip.com

Source (?)

ResNet: ILSVRC 2015 winner

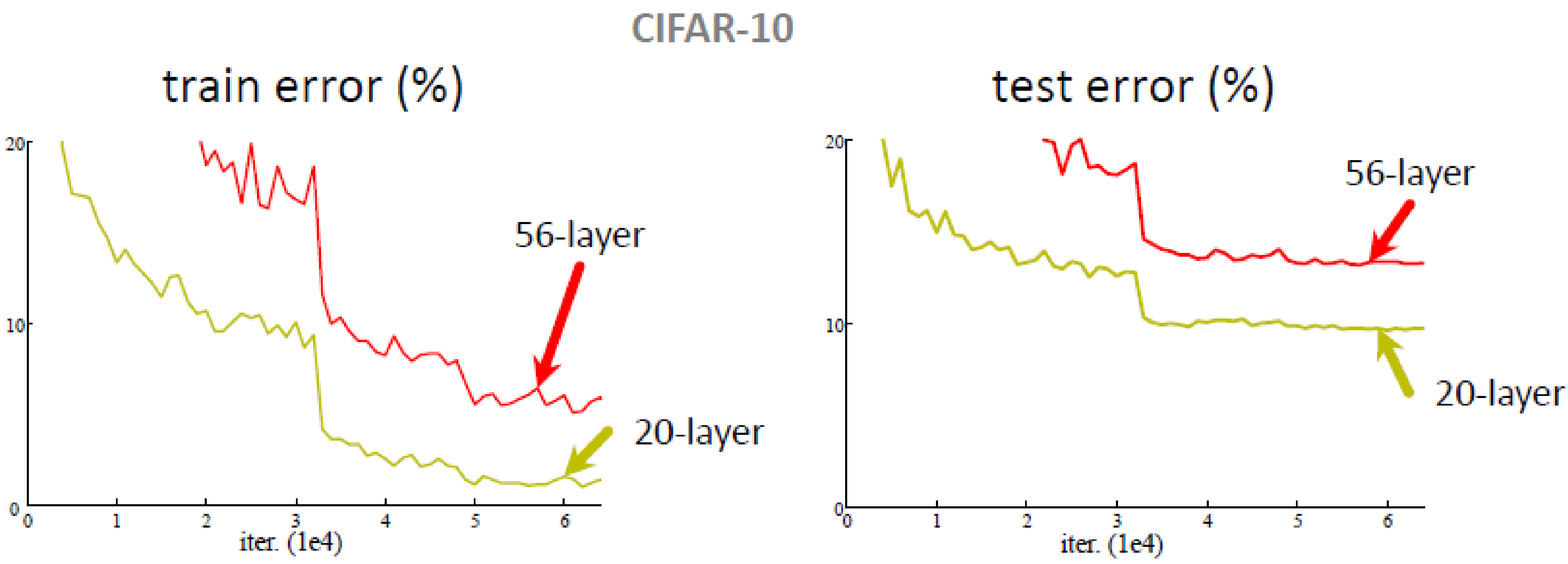


Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, [Deep Residual Learning for Image Recognition], CVPR 2016

Problema:

Até o momento: empilhar convoluções 3x3

Funciona bem até 20~30 camadas



Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

Problema

Para problemas mais complexos

- Campos receptive (convoluções) ↑
- Não lineariedade ↑

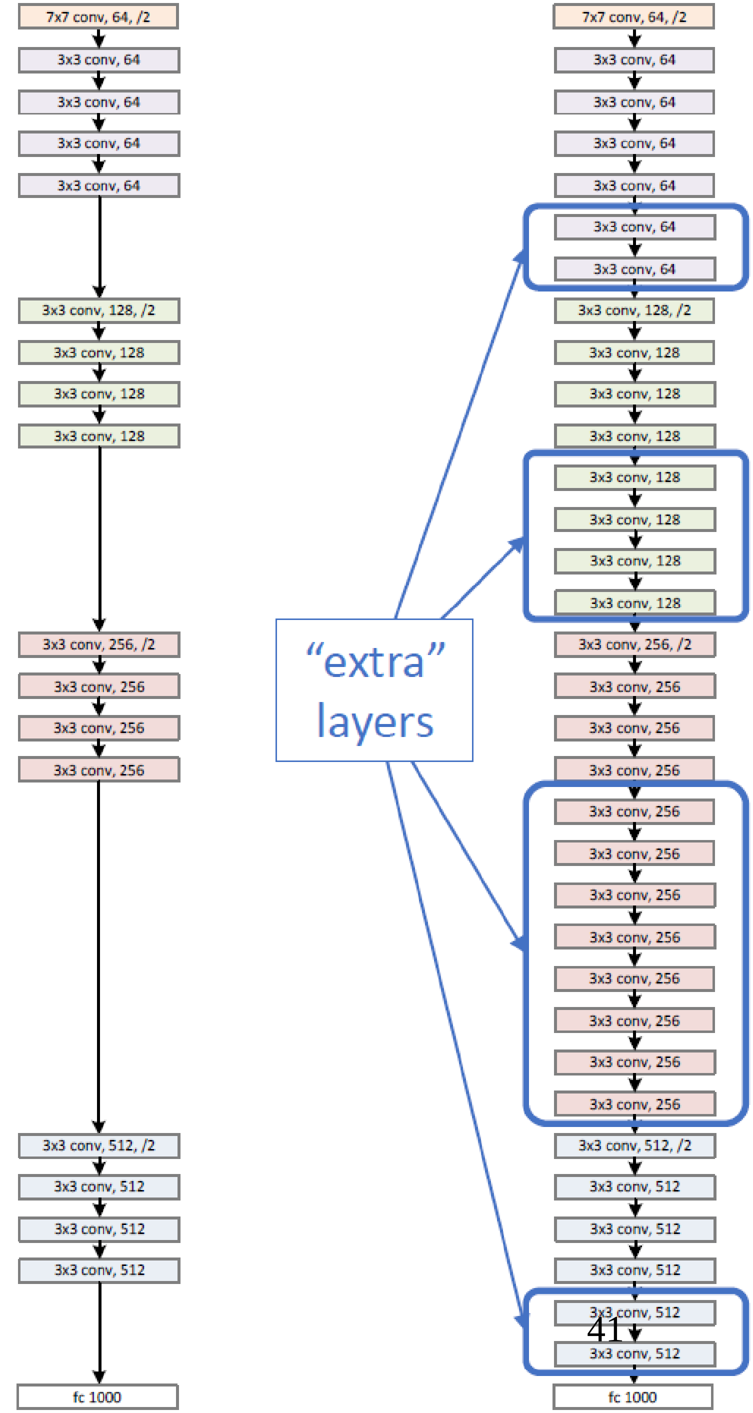
Entretanto, quanto maior a “profundidade”:

- vanishing/exploding gradients problem

ResNet

Solução Naïve

- Se as camadas extras forem um mapeamento de identidade, os erros de treinamento não aumentarão




É MUITA DOIDERA!!!

ResNet

- Módulo residual
  - Introduz conexões de "atalho"

Ideia do neurônio clássico

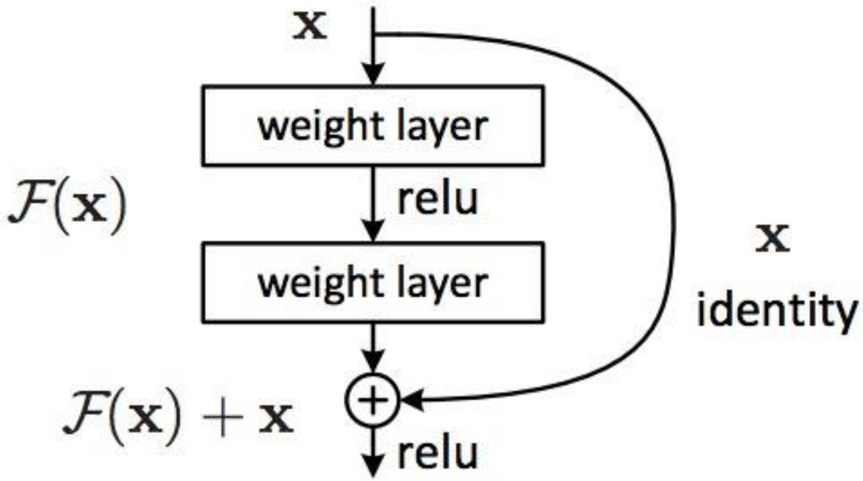
$$x^k = f((W^k * x^{k-1}) + b^k).$$

ResNet

- Módulo residual
  - Introduz conexões de “atalho”

Módulo residual

$$y^k = F(x^k, W^k) + x^k,$$



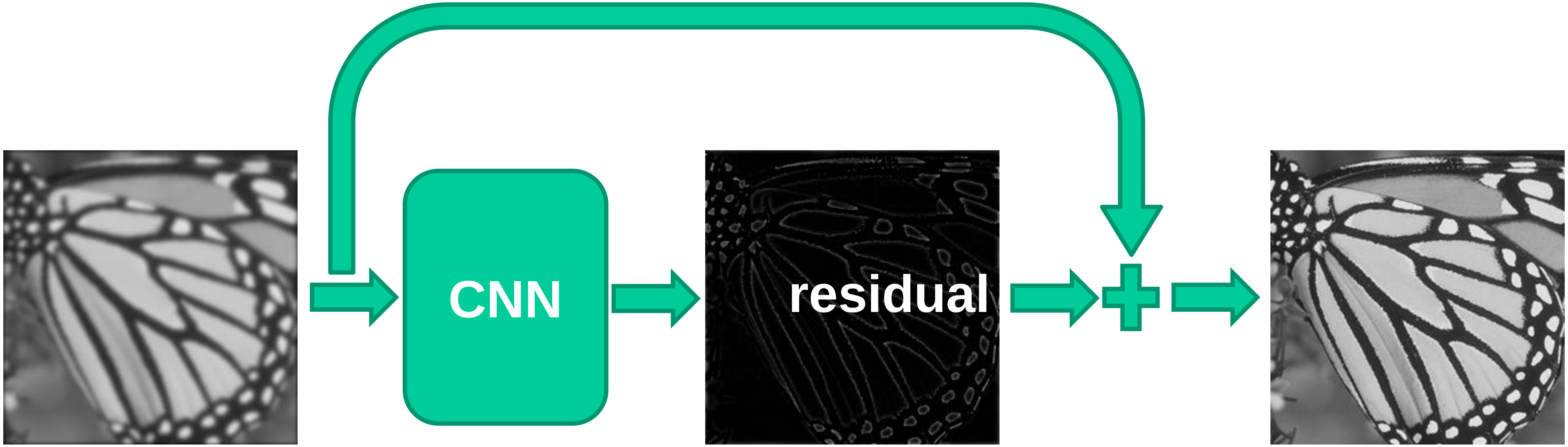
Ideia do neurônio clássico

$$x^k = f((W^k * x^{k-1}) + b^k).$$

ResNet

- Cada pilha de camadas se encaixe diretamente no mapeamento subjacente desejado
- Explicitamente permitimos que essas camadas se encaixem em um mapeamento residual.
- Hipótese de que é mais fácil otimizar o mapeamento residual do que otimizar o mapeamento original.

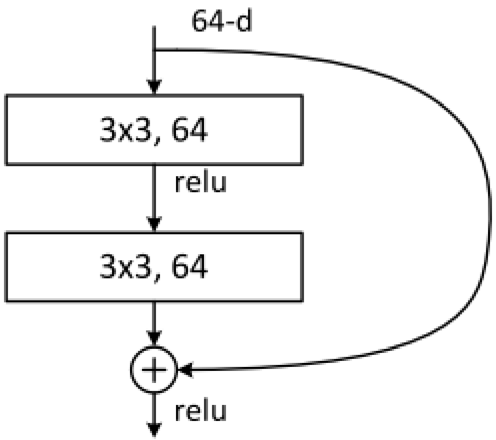
ResNet



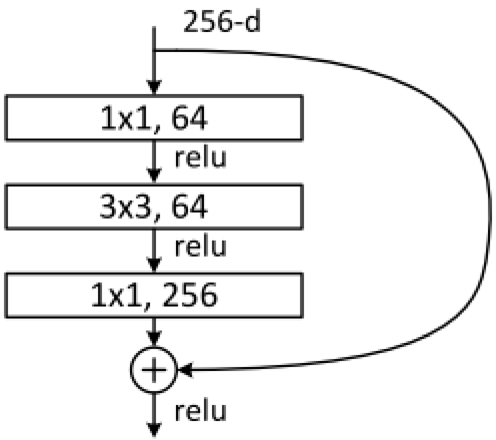
ResNet

Bloco Residual

- Simples
- Livre de parâmetro



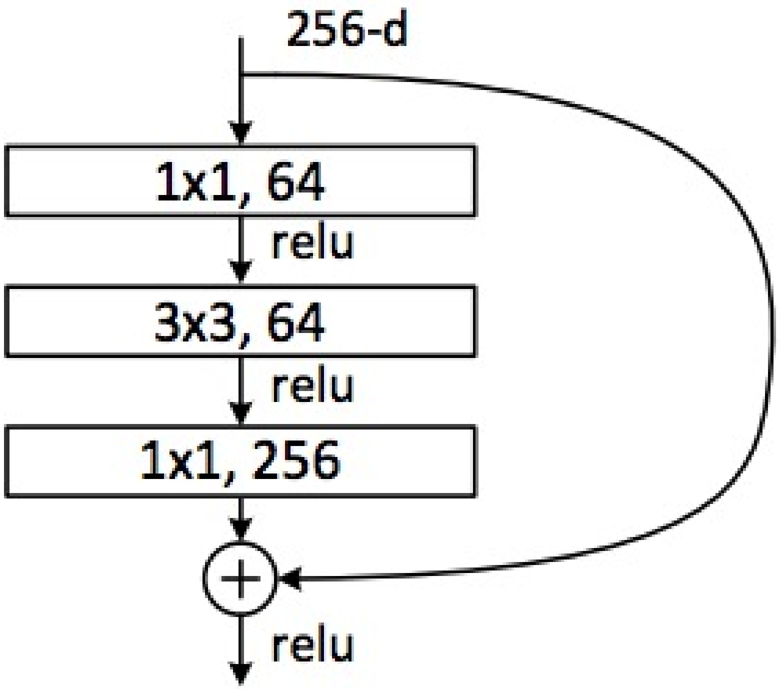
Bloco “naïve”



Bloco “bottleneck” (ResNet-50/101/152)

ResNet

Módulo “bottleneck”



- Com convoluções 3x3 e 256 mapas de entrada:
  256 x 256 x 3 x 3 ~ 600K
- Com convoluções 1x1 para reduzir de 256 para 64 mapas, seguidos por convoluções 3x3, seguidos por convoluções 1x1 para expander para 256 mapas:
  256 x 64 x 1 x 1 ~ 16K
  64 x 64 x 3 x 3 ~ 36K
  64 x 256 x 1 x 1 ~ 16K
  Total: ~70K

ResNet

Alguns tipos de “atalhos”

- Identidade

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}.$$

- Projeção

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}.$$

ResNet

Arquitetura para ImageNet:

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

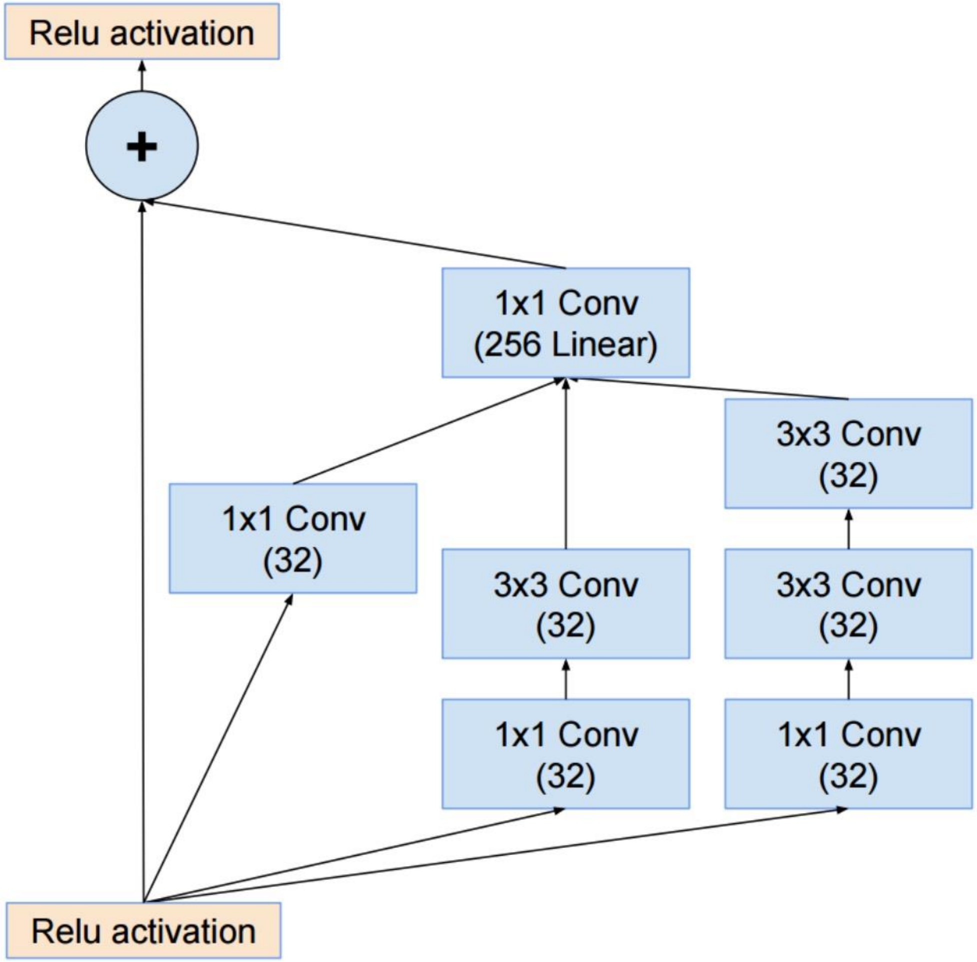
Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, [Deep Residual Learning for Image Recognition](), CVPR 2016 (Best Paper)

Inception v4

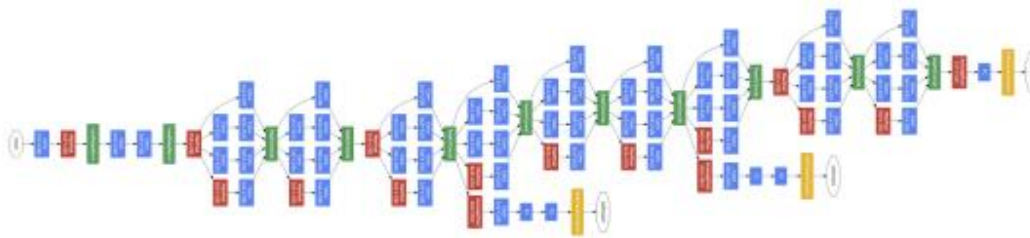
Duas “redes” poderosas:

- **Inception**
- **Residual**

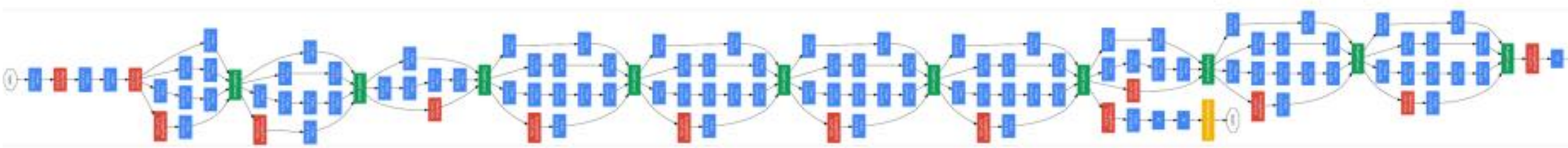
Inception v4



C. Szegedy et al., [Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning], arXiv 2016



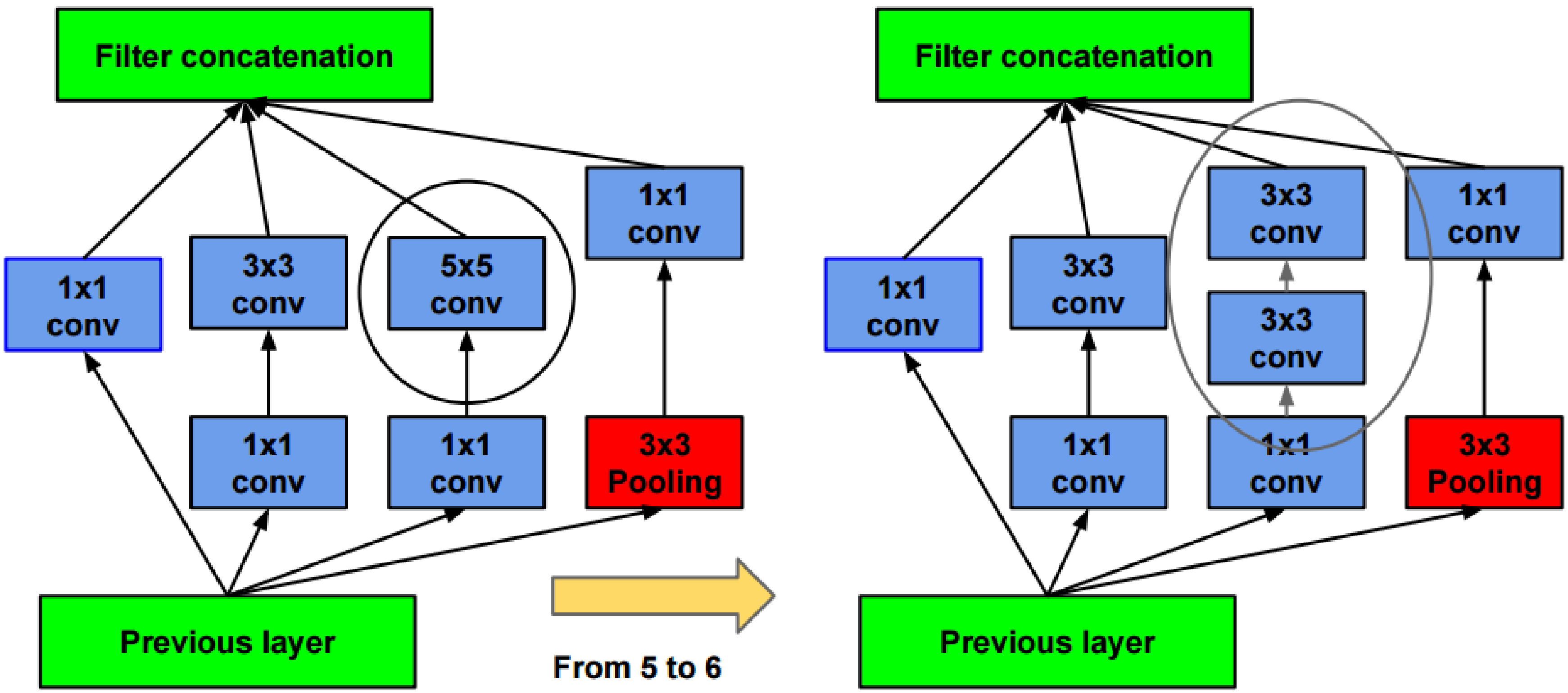
[1]Inception 5 (GoogLeNet)



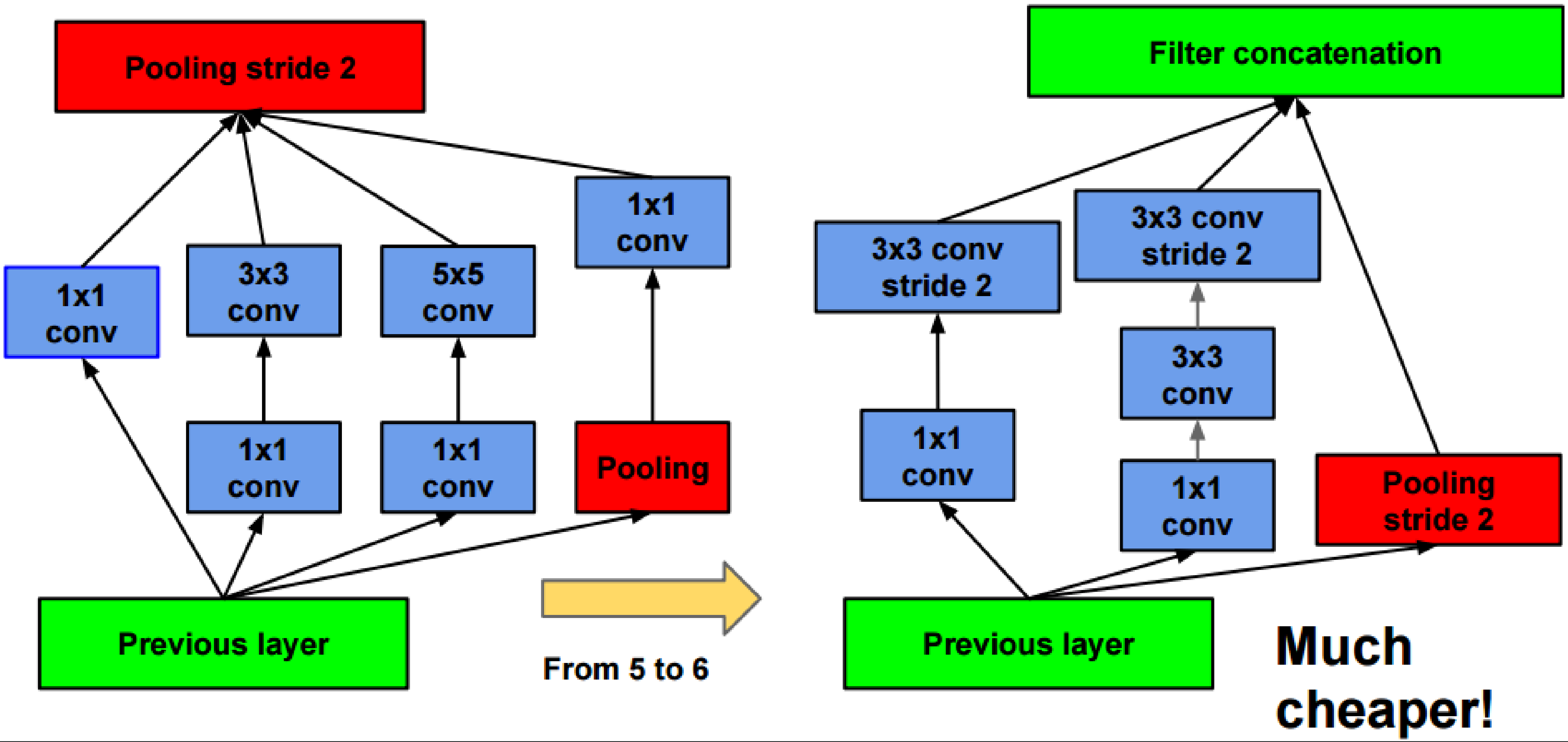
Inception 7a

[1]Going Deeper with Convolutions, [C. Szegedy et al, CVPR 2015]

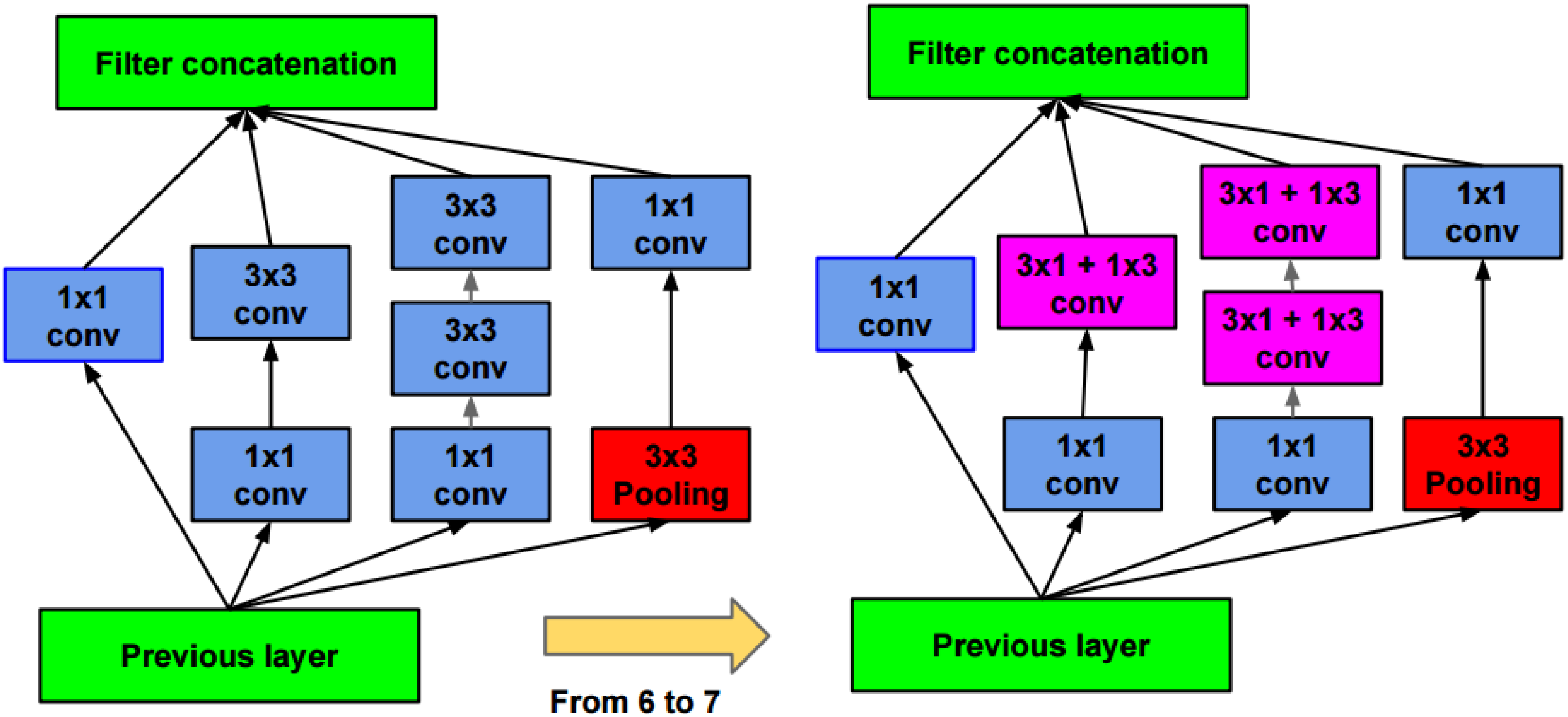
Structural changes from Inception 5 to 6



Grid size reduction Inception 5 vs 6



Structural changes from Inception 6 to 7

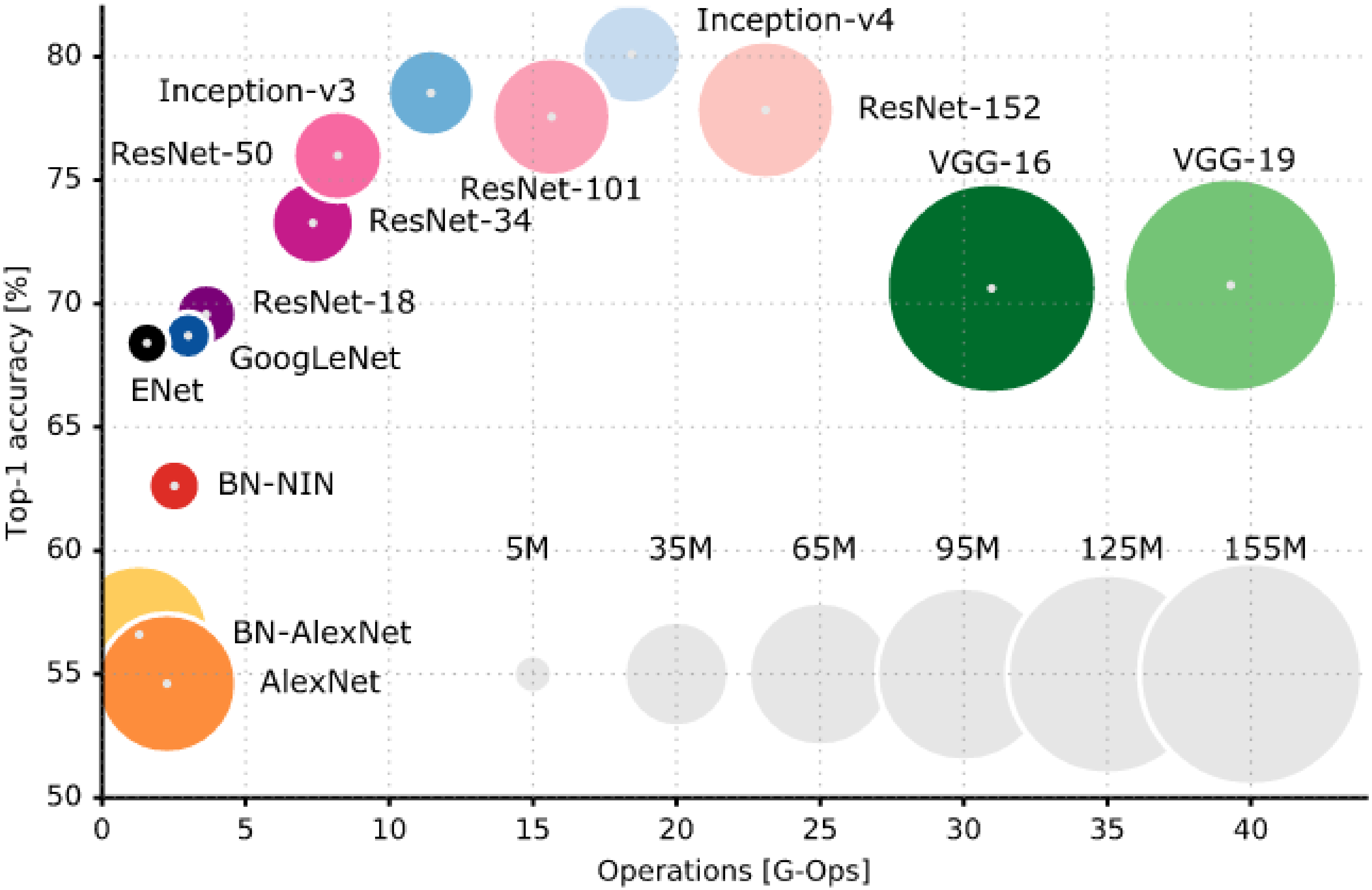


Summary: ILSVRC 2012-2015

Team	Year	Place	Error (top-5)	External data
SuperVision – Toronto (AlexNet, 7 layers)	2012	-	16.4%	no
SuperVision	2012	1st	15.3%	ImageNet 22k
Clarifai – NYU (7 layers)	2013	-	11.7%	no
Clarifai	2013	1st	11.2%	ImageNet 22k
VGG – Oxford (16 layers)	2014	2nd	7.32%	no
GoogLeNet (19 layers)	2014	1st	6.67%	no
ResNet (152 layers)	2015	1st	3.57%	
Human expert*			5.1%	

http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/

Accuracy vs. efficiency



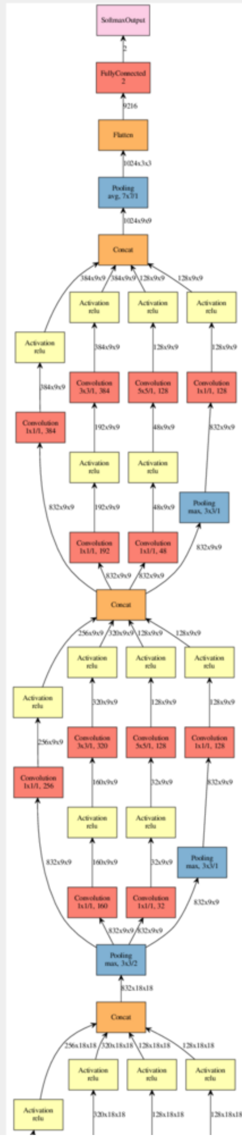
https://culurciello.github.io/tech/2016/06/04/nets.html

LeNet

AlexNet

VGG

GoogLeNet



Inception V3

Inception BN


LeNet
AlexNet
VGG
GoogLeNet
Inception V3
Inception BN
Puta que Pariu
Ia morrer sem saber disso.

Highway Networks

Para uma camada densa “convencional”

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W_H})$$

```
def dense(x, input_size, output_size, activation):
  W = tf.Variable(tf.truncated_normal([input_size, output_size], stddev=0.1), name="weight")
  b = tf.Variable(tf.constant(0.1, shape=[output_size]), name="bias")
  y = activation(tf.matmul(x, W) + b)
  return y
```

Highway Networks

A camada tem dois gateways que controlam o fluxo de informações. O gate que controla o quanto da ativação passa para a próxima camada... e o gate que controla o quanto da entrada sem modificação passa "direto" para a próxima camada.

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W_H}) \cdot T(\mathbf{x}, \mathbf{W_T}) + \mathbf{x} \cdot C(\mathbf{x}, \mathbf{W_C})$$

- Um conjunto extra de pesos e bias para aprender os “gates”
- A operação de transformção (***T***).
- “carry gate” (***C*** equivalente a ***1 - T***).

Highway Networks

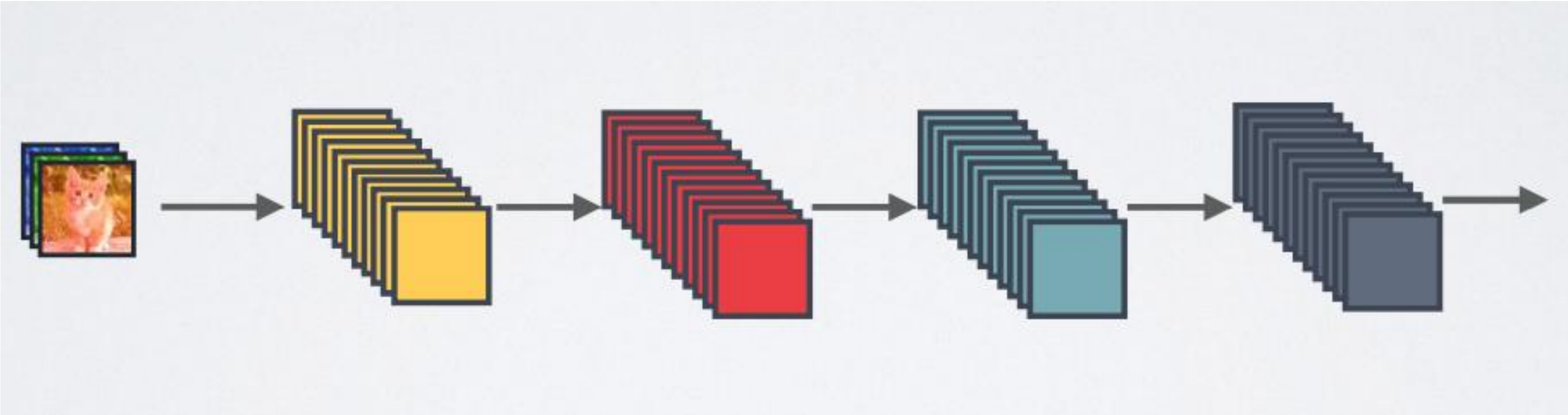
Para saber mais...

Training Very Deep Networks

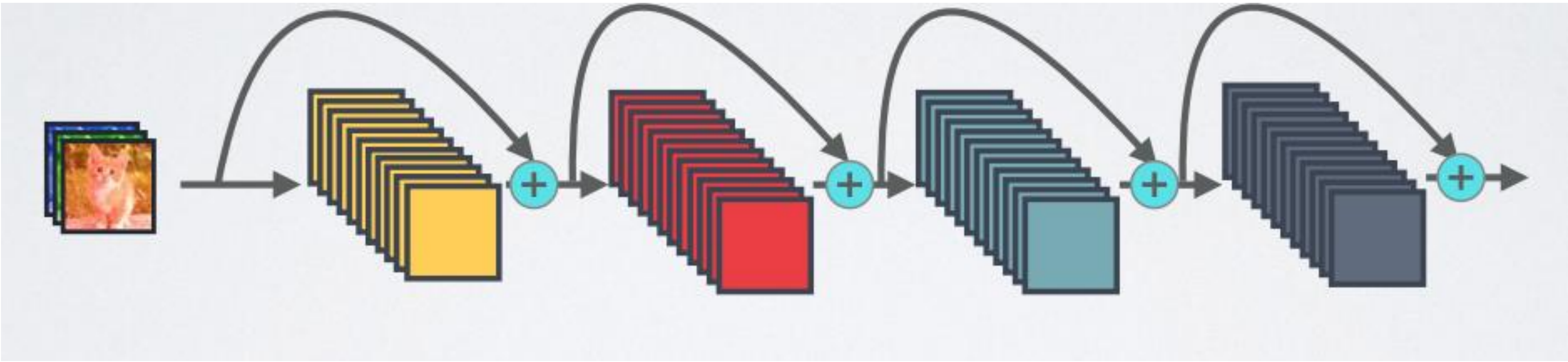
Rupesh Kumar Srivastava   Klaus Greff   Jürgen Schmidhuber

The Swiss AI Lab IDSIA / USI / SUPSI
{rupesh, klaus, juergen}@idsia.ch

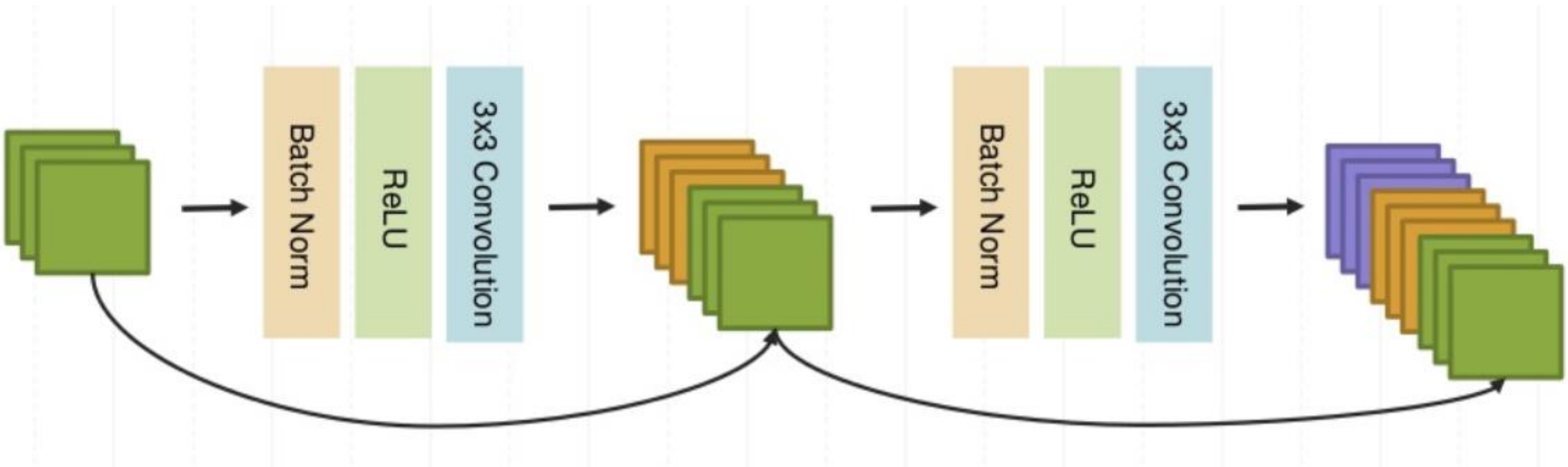
Densenet

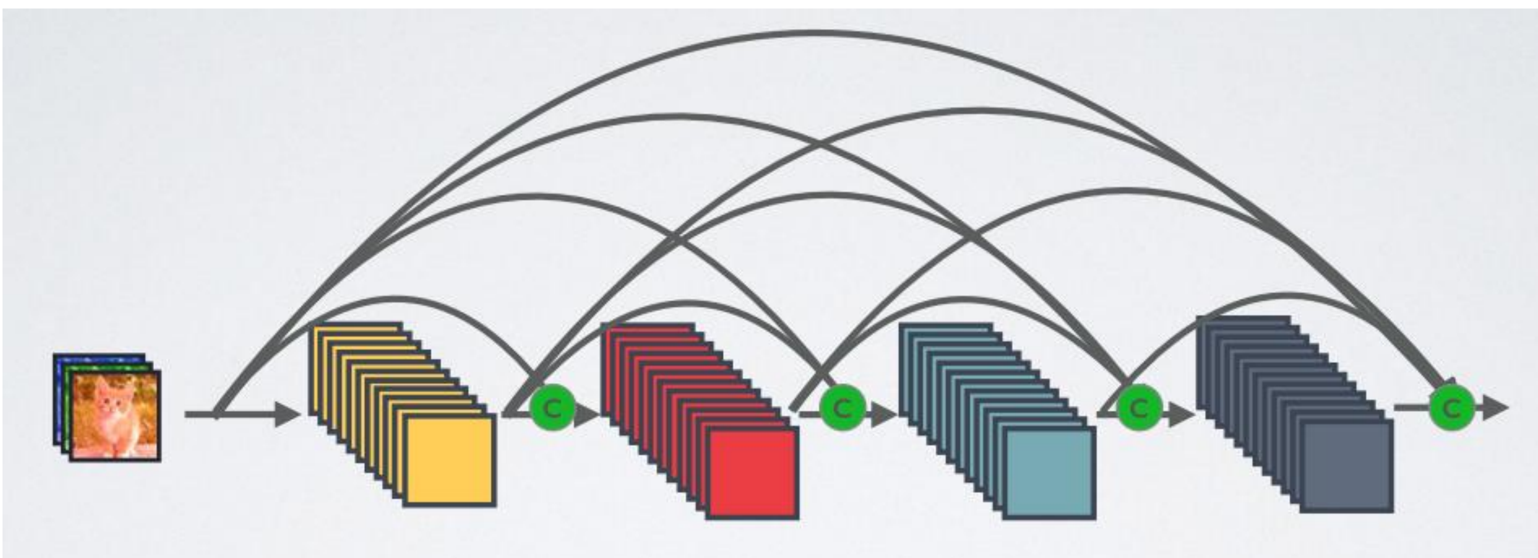


Densenet

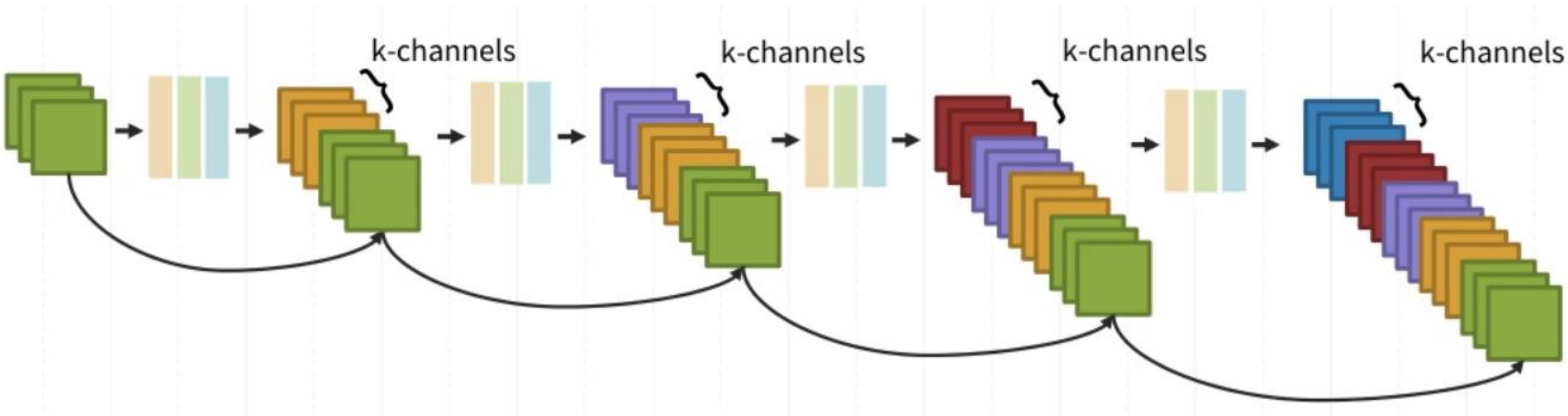


Densenet

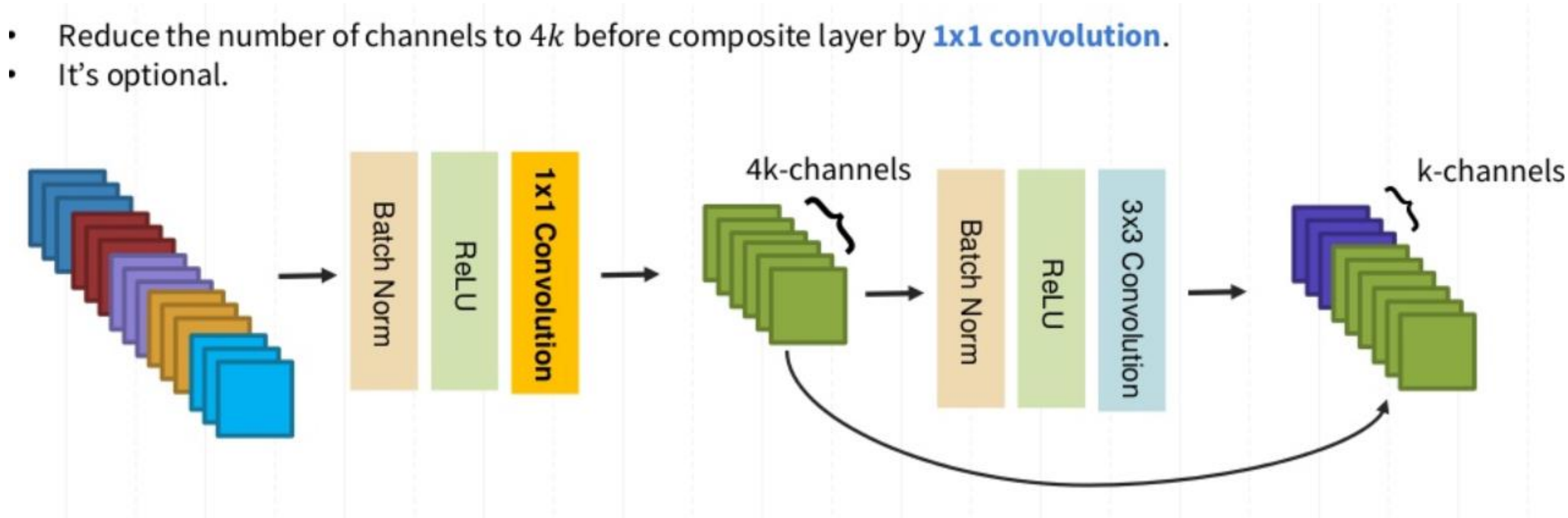

Batch Norm
ReLU
3x3 Convolution
Batch Norm
ReLU
3x3 Convolution



Densenet



Densenet



Reduce the number of channels to 4k before composite layer by 1x1 convolution.
It's optional.
Batch Norm
ReLU
1x1 Convolution
4k-channels
Batch Norm
ReLU
3x3 Convolution
k-channels

Transfer Learning

Transfer Learning

Gato

Cachorro

elephant

tige
r

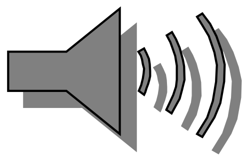
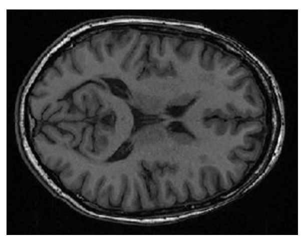
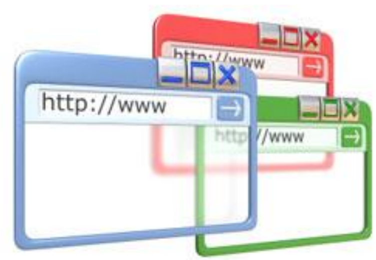
dog

cat

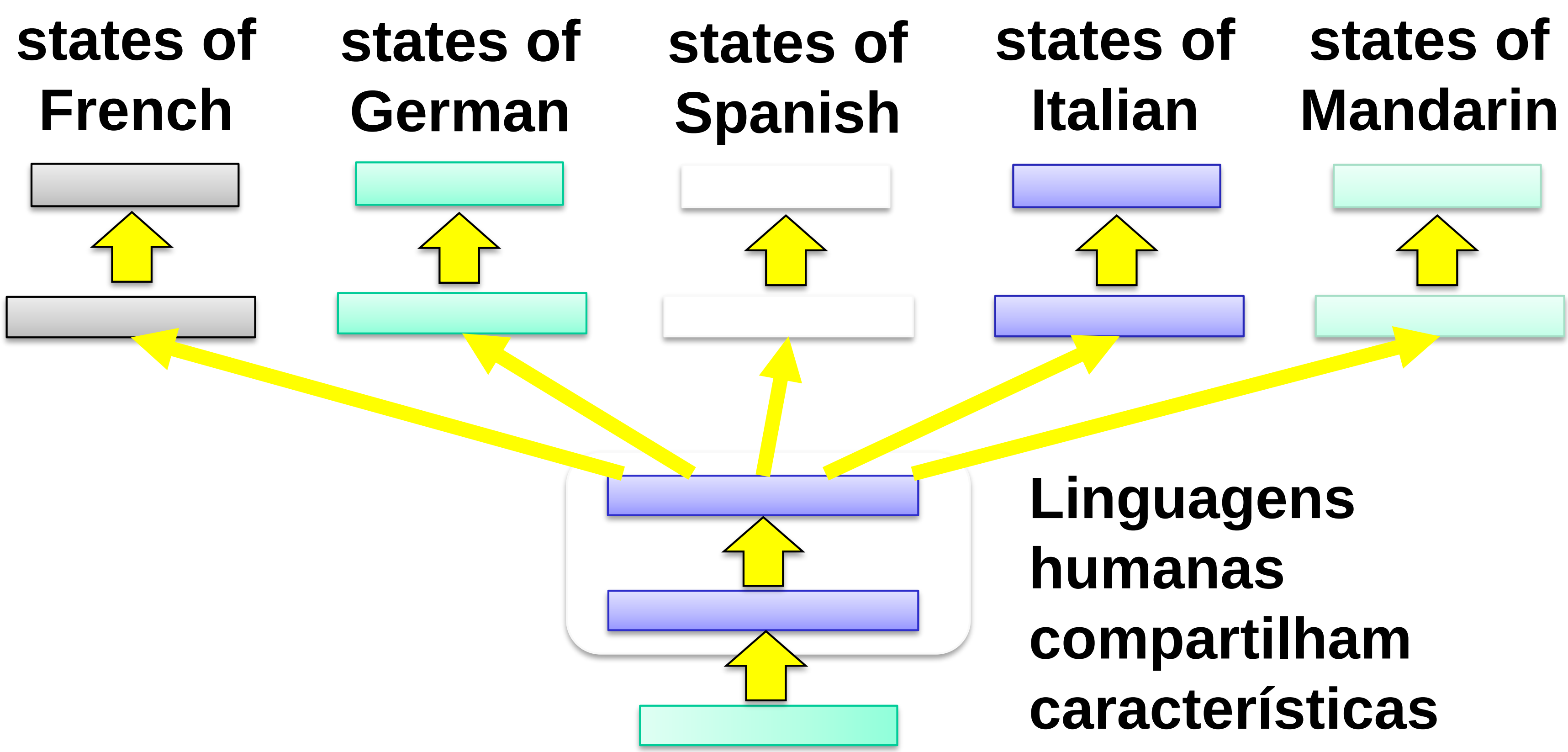
Domínio similar, tarefas diferentes

Domínio diferente, mesma tarefa

Transfer Learning

	Tarefa	Dados
Speech Recognition	 Taiwanese	English Chinese ……
Image Recognition	 Medical Images	
Text Analysis	 Specific domain	 Webpag es

Transfer Learning



<u>Similar idea in translation</u>: **Daxiang Dong, Hua Wu, Wei He, Dianhai Yu and Haifeng Wang, "Multi-task learning for multiple language translation.“, ACL 2015**

Transfer Learning Model Fine-tuning

Tarefa

- Source data: (x^s, y^s) ⇦ **Muito**
- Target data: (x^t, y^t) ⇦ **Pouco**

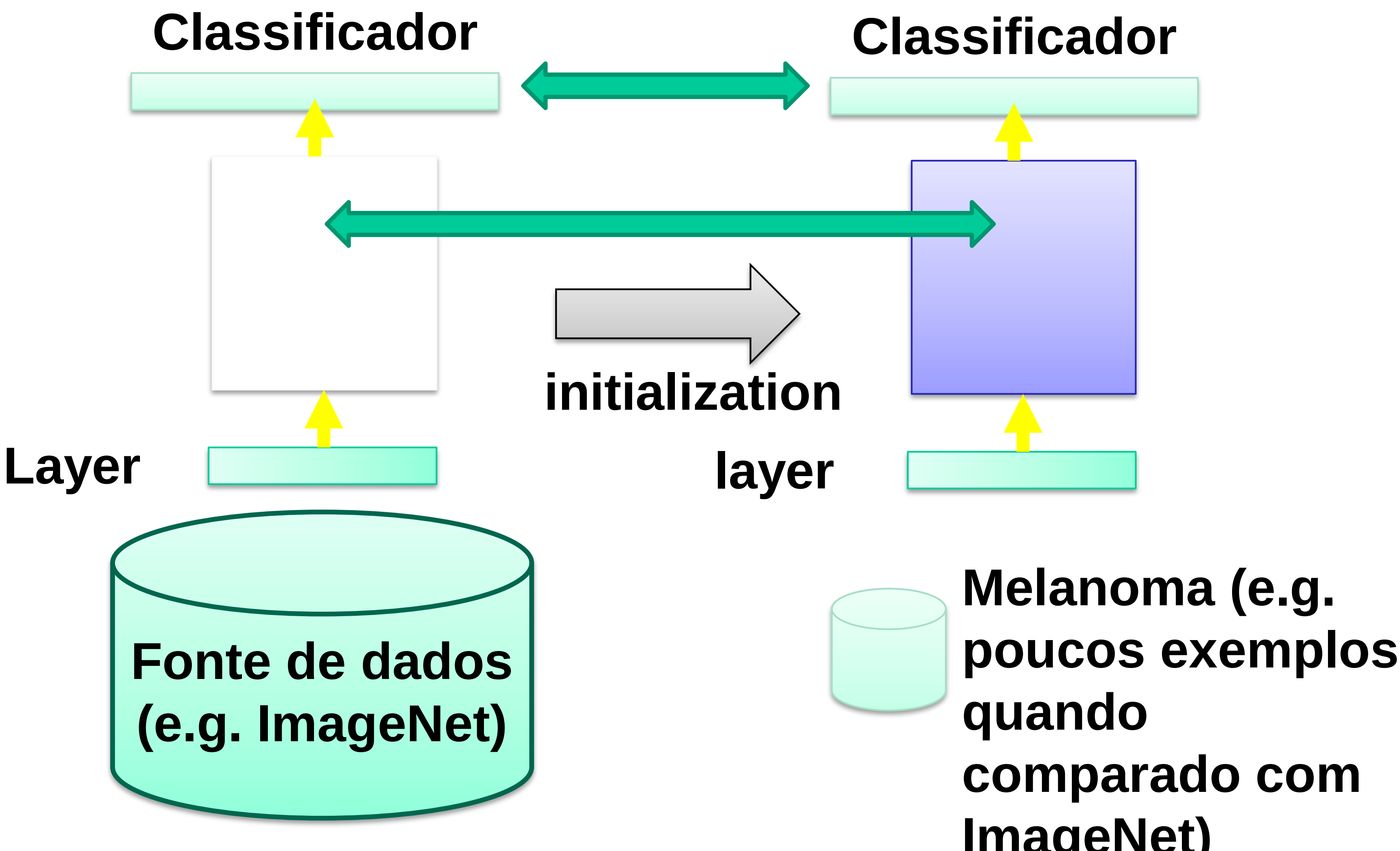
Exemplos: reconhecimento de fala e melanoma

- Fonte de dados: audio e transcrição de várias falas
- Alvo: transcrição para texto

Idea: Treinar um modelo a partir de uma fonte grande de dados, depois fazer um ajuste fino para a situação em que se tem poucos dados

- Desafio: cuidado com o overfitting

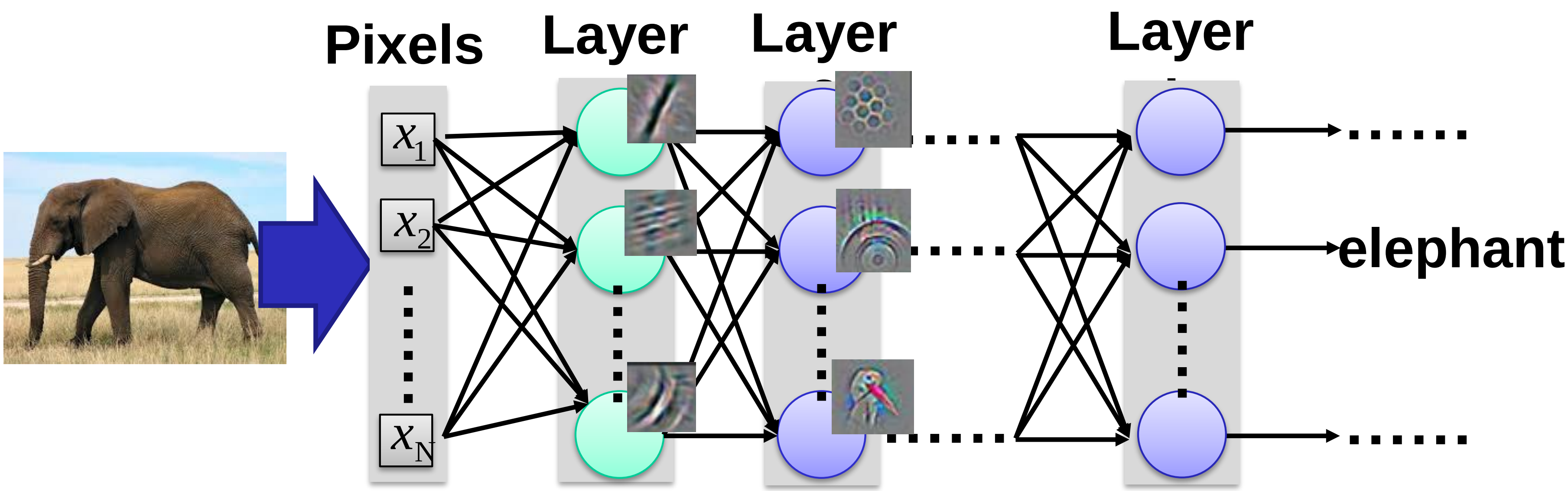
Conservative Training



Layer Transfer

Quais camadas podem ser transferidas?

- PLN: Geralmente as últimas camadas
- Imagem: Geralmente as primeiras camadas

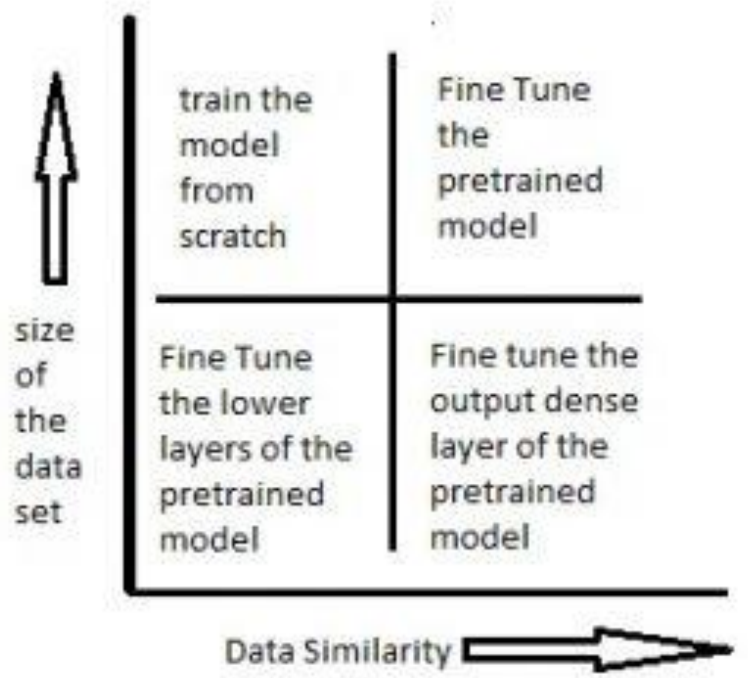


Transfer Learning - Overview

		Source Data	
		supervisionado	Não supervisionado
Target Data	Superv.	Fine-tuning Multitask Learning	**Self-taught learning** Rajat Raina , Alexis Battle , Honglak Lee , Benjamin Packer , Andrew Y. Ng, Self-taught learning: transfer learning from unlabeled data, ICML, 2007
	Não Super.	Domain-adversarial training Zero-shot learning	**Self-taught Clustering** **Wenyuan Dai, Qiang Yang, Gui-Rong Xue, Yong Yu, "Self-taught clustering", ICML 2008**

Transfer Learning

- Extração de características: retreinar apenas o classificador (Fully Coneceted Layer)
- Aproveitar somente a arquitetura
- Reinicializar algumas camadas



Transfer Learning

- **Cenário 1 – Data set pequeno e a similaridade dos dados é muito alta**
- Use a rede pretreinada como feature extractor.
- Exemplo: Suponha que vamos usar uma rede treinada para um novo conjunto de gatos e cachorros.
- ImageNet: 1000 classes de saída
- Modificamos a últimas camadas (dense layers) e a softmax para duas categorias

Transfer Learning

- **Cenário 2 – Data set pequeno e similaridade de dados baixa**
- Mantenha as camadas iniciais da rede pretreinada.
- Retreine as camadas intermediárias e finais

Transfer Learning

- **Cenário 3 – Data set grande e similaridade de dados baixa**
- Neste caso…. Fuja para as montanhas…

Transfer Learning

- **Cenário 4 – Data set grande e alta similaridade**
- Situação ideal: no máximo treinar o classificador no topo da rede.

Bônus

Transferência de estilo

A neural Alogirithm of Artistic Style

https://arxiv.org/pdf/1508.06576.pdf

Entrada **Imagem de estilo**

Transferência de estilo

$\boxed{\text{Perceptual Loss}}$ **for Real-Time Style transfer and Super-Resolution**

↕

Per-Pixel Loss : y-y^

Transferência de estilo

Feature Reconstruction Loss

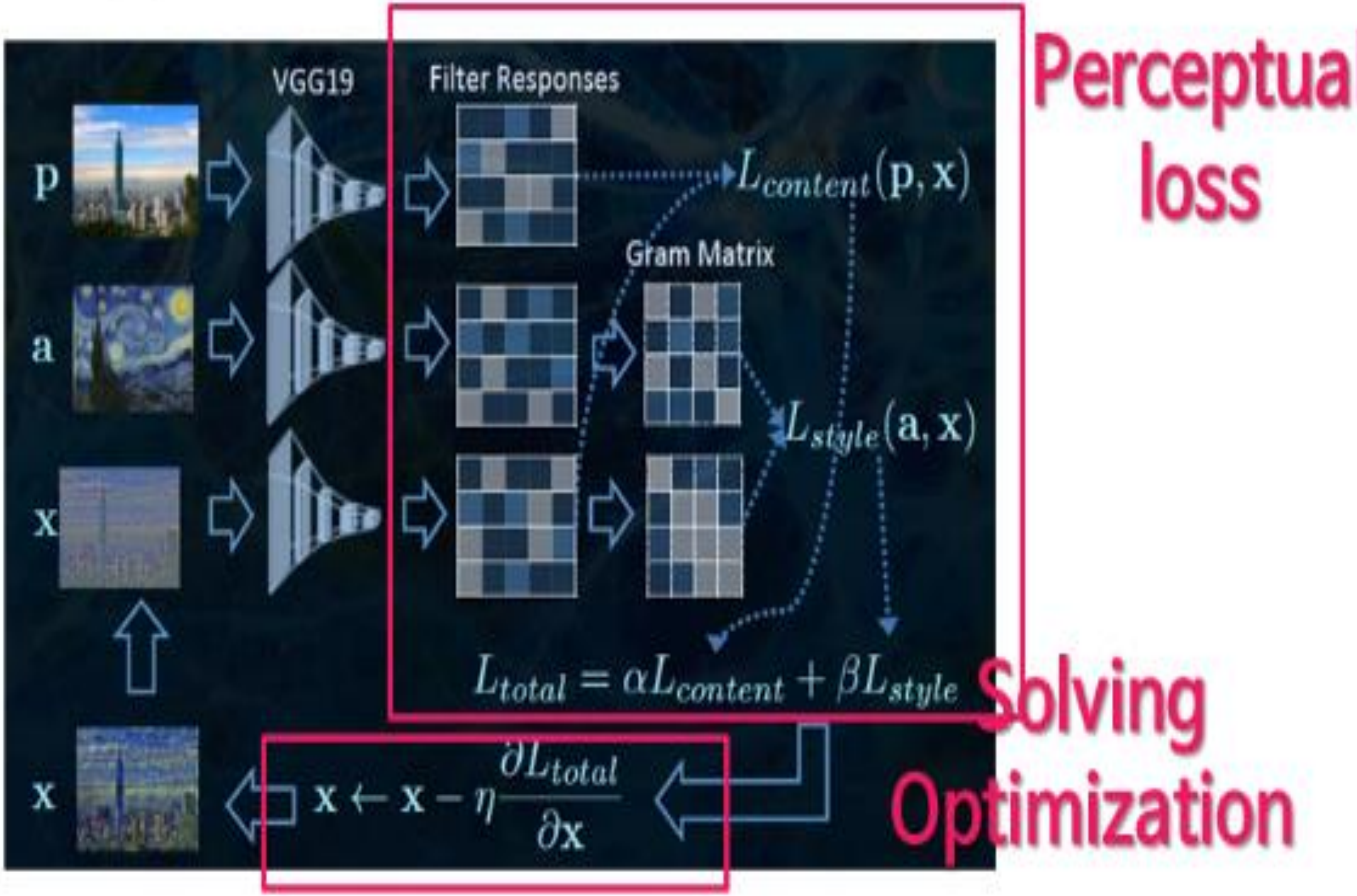
$$\ell_{feat}^{\phi,j}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \|\phi_j(\hat{y}) - \phi_j(y)\|_2^2$$

+

Style Reconstruction Loss

$$\ell_{style}^{\phi,j}(\hat{y}, y) = \|G_j^{\phi}(\hat{y}) - G_j^{\phi}(y)\|_F^2.$$

Transferência de estilo



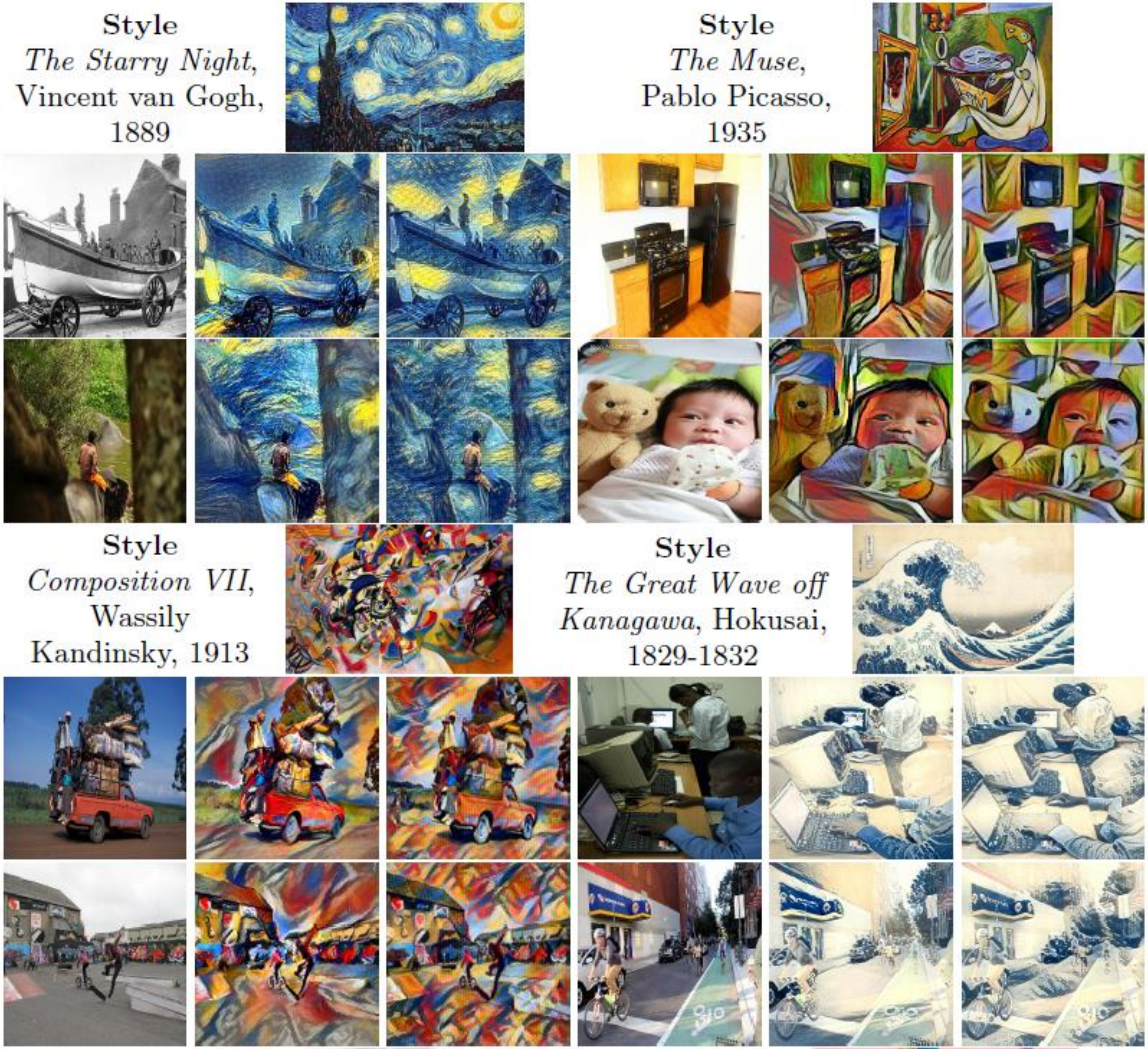
Transferência de estilo

Conceito de "reconstrução de estilo“: Em vez de tentar combinar as ativações, tente combinar a correlação das ativações.

O recurso de correlação é chamado de matriz de Gram: produto de ponto entre a matriz de ativação de característica vetorial e sua transposição.

A definição da matriz de Gram pode parecer confusa. Tradução literal da equação 3 do artigo original usando numpy:

```
def gram(layer):
  N = layer.shape[1]
  F = layer.reshape(N, -1)
  M = F.shape[1]
  G = np.zeros((N, N))
  for i in range(N):
    for j in range(N):
      for k in range(M):
        G[i,j] += F[i,k] * F[j,k]
  return G
```


Style
The Starry Night, Vincent van Gogh, 1889
Style
The Muse, Pablo Picasso, 1935
Style
Composition VII, Wassily Kandinsky, 1913
Style
The Great Wave off Kanagawa, Hokusai, 1829-1832

Lista de leituras

- https://culurciello.github.io/tech/2016/06/04/nets.html
- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86(11): 2278–2324, 1998.
- A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012
- M. Zeiler and R. Fergus, Visualizing and Understanding Convolutional Networks, ECCV 2014
- K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015
- M. Lin, Q. Chen, and S. Yan, Network in network, ICLR 2014
- C. Szegedy et al., Going deeper with convolutions, CVPR 2015
- C. Szegedy et al., Rethinking the inception architecture for computer vision, CVPR 2016
- K. He, X. Zhang, S. Ren, and J. Sun, Deep Residual Learning for Image Recognition, CVPR 2016

https://datascience.stackexchange.com/questions/15328/what-is-the-difference-between-inception-v2-and-inception-v3

https://pdfs.semanticscholar.org/73ac/009051bba99eaea799172b28d69168b6aa02.pdf

http://blog.csdn.net/u010025211/article/details/51206237

http://lsun.cs.princeton.edu/slides/Christian.pdf

https://sourcedexter.com/retrain-tensorflow-inception-model/

https://www.quora.com/How-are-1x1-convolutions-used-for-dimensionality-reduction

http://www.ic.unicamp.br/~rocha/teaching/2015s2/mo826/classes/2015-google-faces.pdf

http://www.cv-foundation.org/openaccess/content_cvpr_2015/papers/Szegedy_Going_Deeper_With_2015_CVPR_paper.pdf

http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture9.pdf

https://www.robots.ox.ac.uk/~vgg/rg/slides/rg_4feb16.pdf