Segmentação Semântica

Prof. Jefersson A. dos Santos

jefersson@dcc.ufmg.br

Roteiro

Aulas anteriores

- Regressão
- Otimização
- Redes em múltiplas camadas
- Redes convolucionais
- …

Próximas aulas

- Aplicações de CNNs
  - Segmentação semântica
  - Detecção de objetos
  - Reconhecimento em cenário aberto
  - Fundamentos de PDI

Roteiro

Aula de hoje

- Segmentação semântica como classificação de pixels
- FCNs
  - Upsampling e Learnable Sampling
- U-Nets
- SegNets
- Convoluções dilatadas

Redes Convolucionais

Convolutional Neural Networks (CNNs)

This image is CC0 public domain

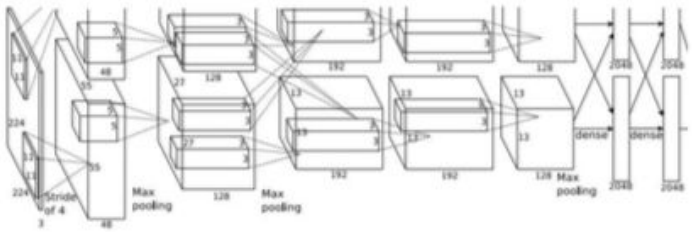


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector:
4096

Fully-Connected:
4096 to 1000

Redes Convolucionais

Convolutional Neural Networks (CNNs)

This image is CC0 public domain

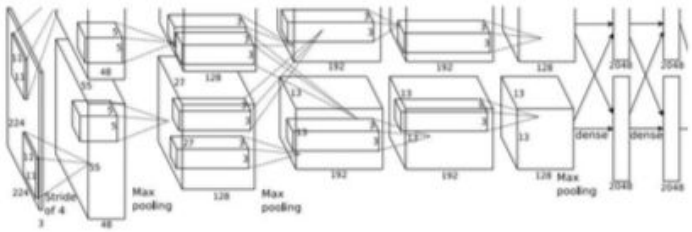


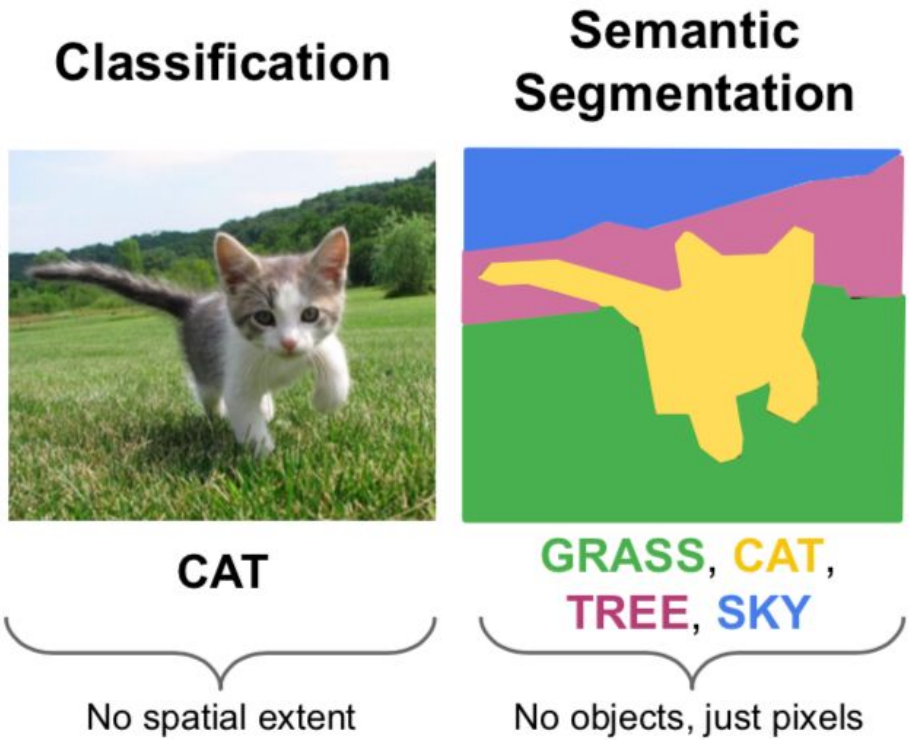
Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

Vector:
4096

Fully-Connected:
4096 to 1000

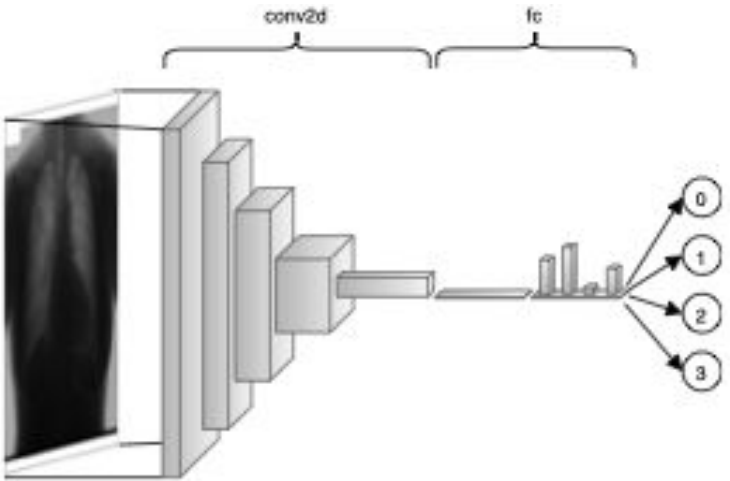
Class Scores
Cat: 0.9
Dog: 0.05
Car: 0.01
...

Aplicações de Redes Convolucionais



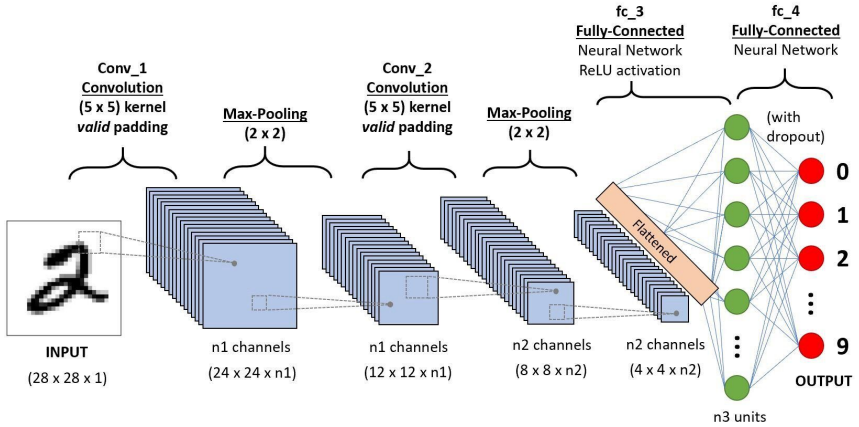
Predição esparsa

- Classificação é um problema de **predição esparsa**, pois queremos atribuir um rótulo a uma imagem
    - CNN é a arquitetura mais usada em imagens



Predição esparsa

- Principais camadas
  - Camadas convolucionais
  - Camadas de Max-Pooling
  - Camadas Fully-Connected (FC)
- Outras camadas/funções
  - Rectified Linear Unit (ReLU)
  - Batch normalization
  - Softmax para classificação
  - Sigmóide para regressão



Rotulação Esparsa → Rotulação Densa

- Rotulação esparsa
  - Classificação de objetos
  - Regressão da probabilidade de malignidade em imagens radiológicas
  - Classificação entre imagens de plantação de café e de imagens urbanas

- Rotulação densa
  - Segmentação de objetos
  - Segmentação de tumores
  - Segmentação de regiões de plantação

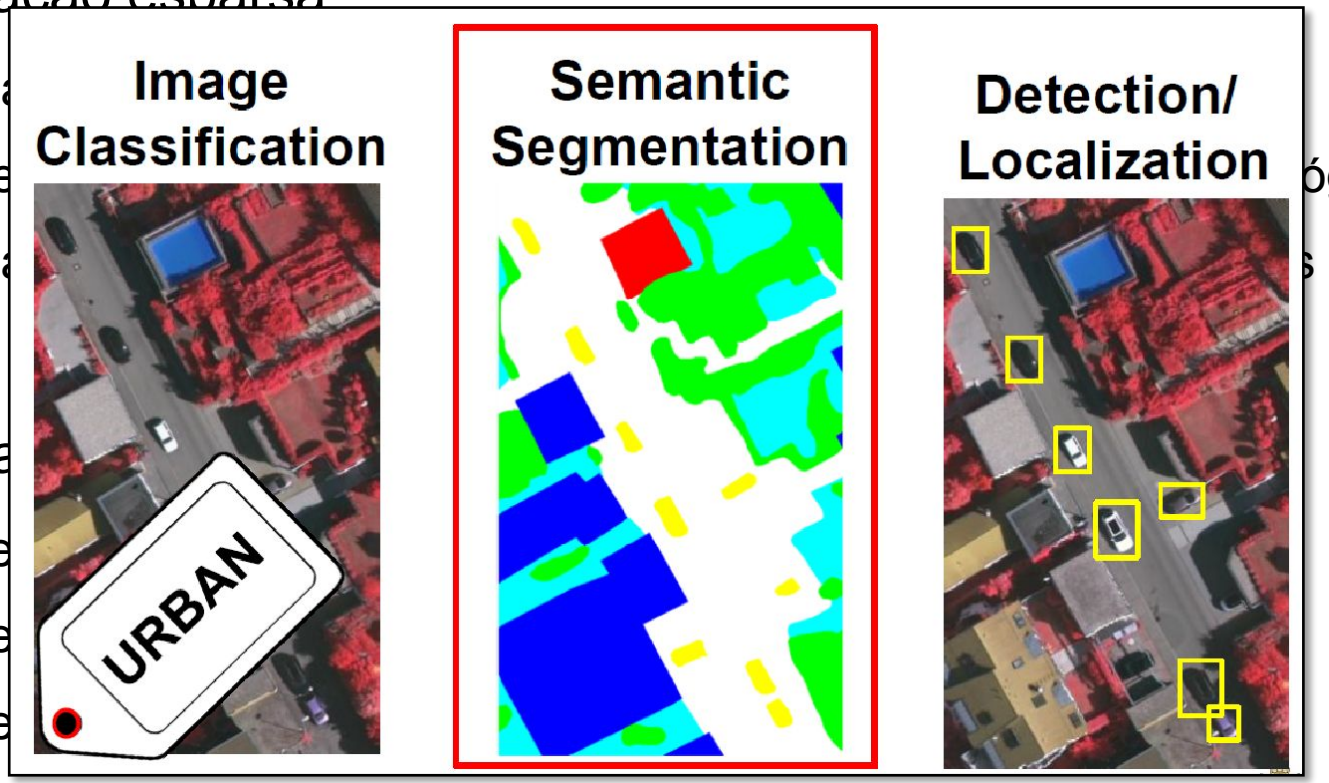
Rotulação Esparsa → Rotulação Densa

- Rotulação esparsa
  - Cl
  - Re
  - Cl

ógicas

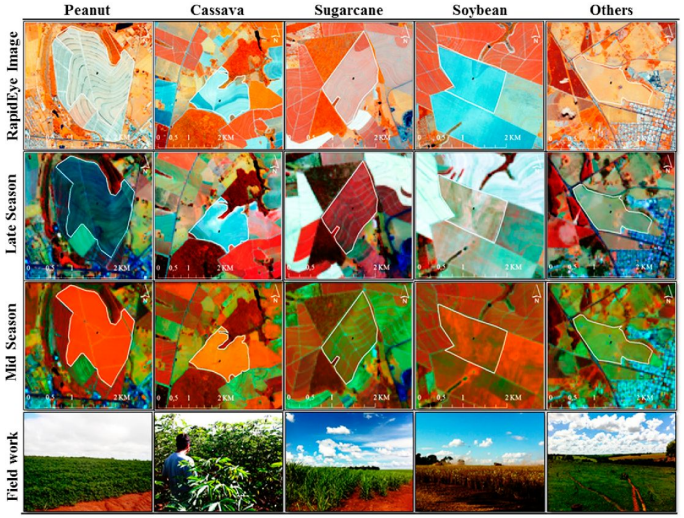
s urbanas

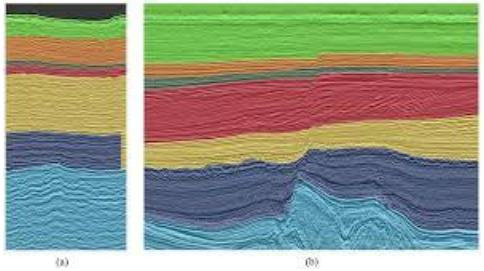
- Rotula
  - Se
  - Se
  - Se

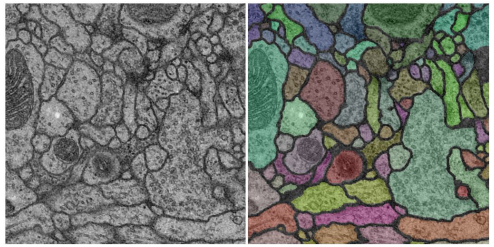


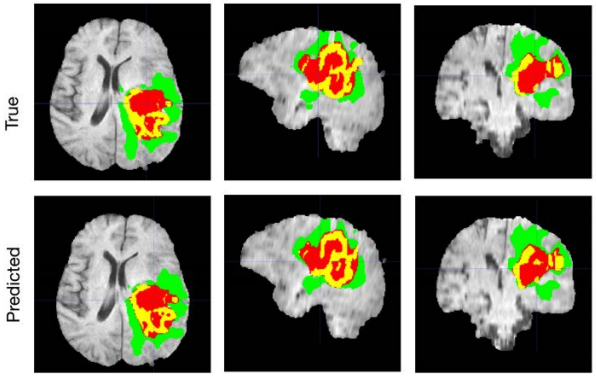
Aplicações

- Diagnóstico médico
- Mapeamento geográfico
- Geologia
  - Mapeamento de lavra (mineração)
  - Exploração de petróleo (sísmica)
- ...







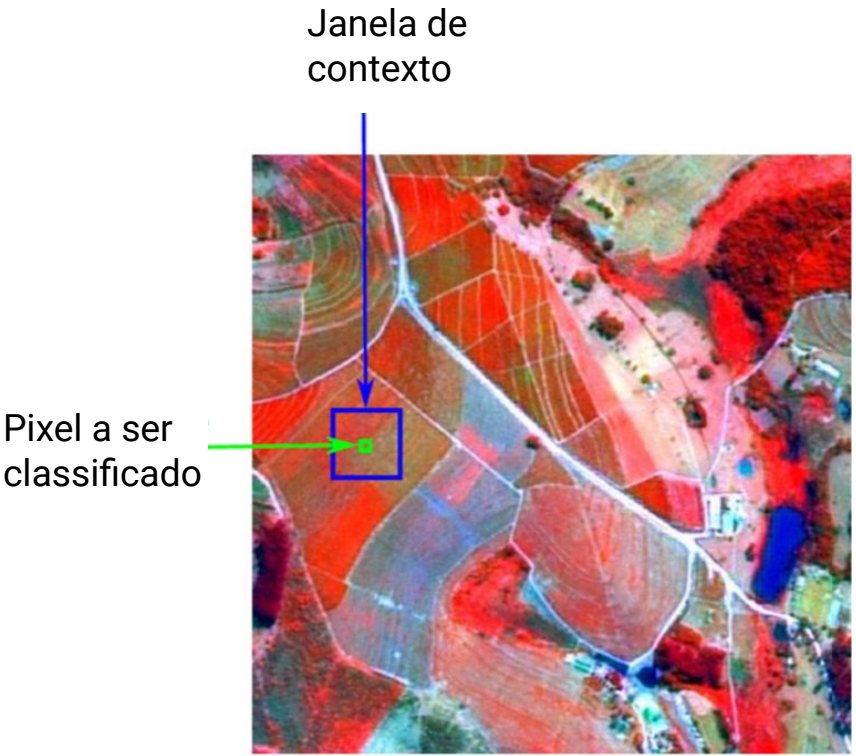


Abordagens profundas para segmentação semântica

Abordagens profundas de segmentação

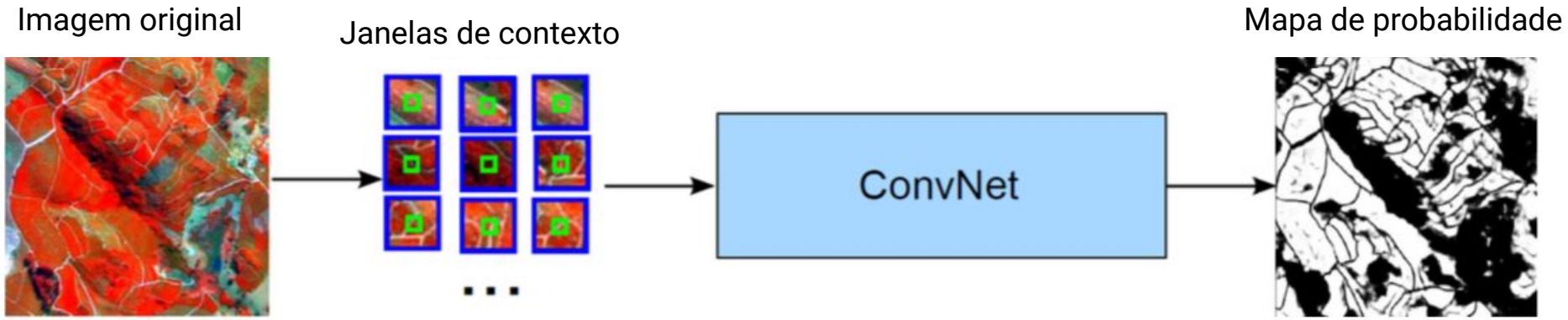
- Abordagens profundas para rotulação densa
  - Classificação de pixels
  - Fully Convolutional Networks (FCNs)
  - Redes de Deconvolução
    - DeconvNets
    - U-Nets
    - SegNets

Classificação de pixels



Nogueira et al. Learning to semantically segment high-resolution remote sensing images. ICPR 2016.

Classificação de pixels



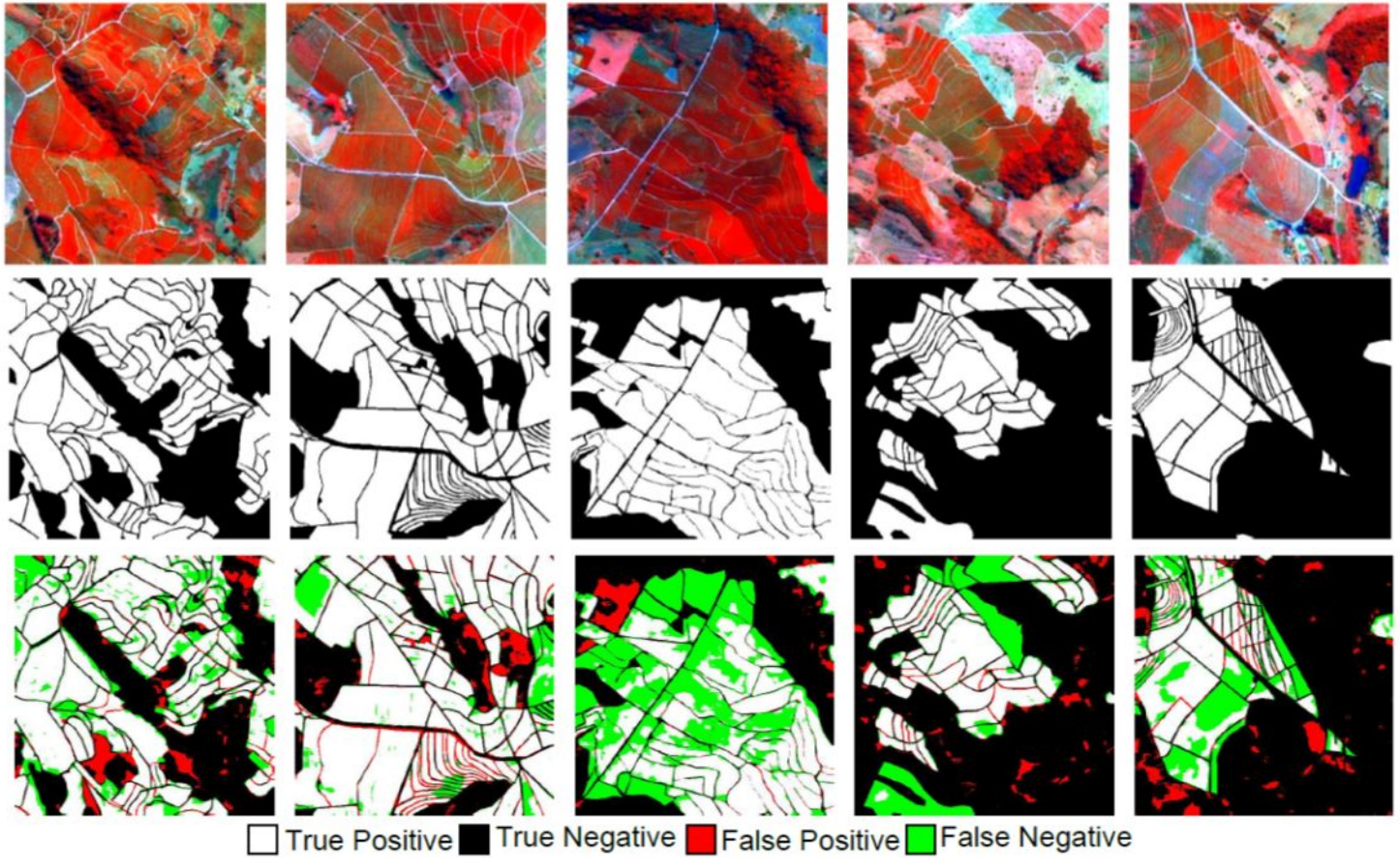
Nogueira et al. Learning to semantically segment high-resolution remote sensing images. ICPR 2016.

Resultados

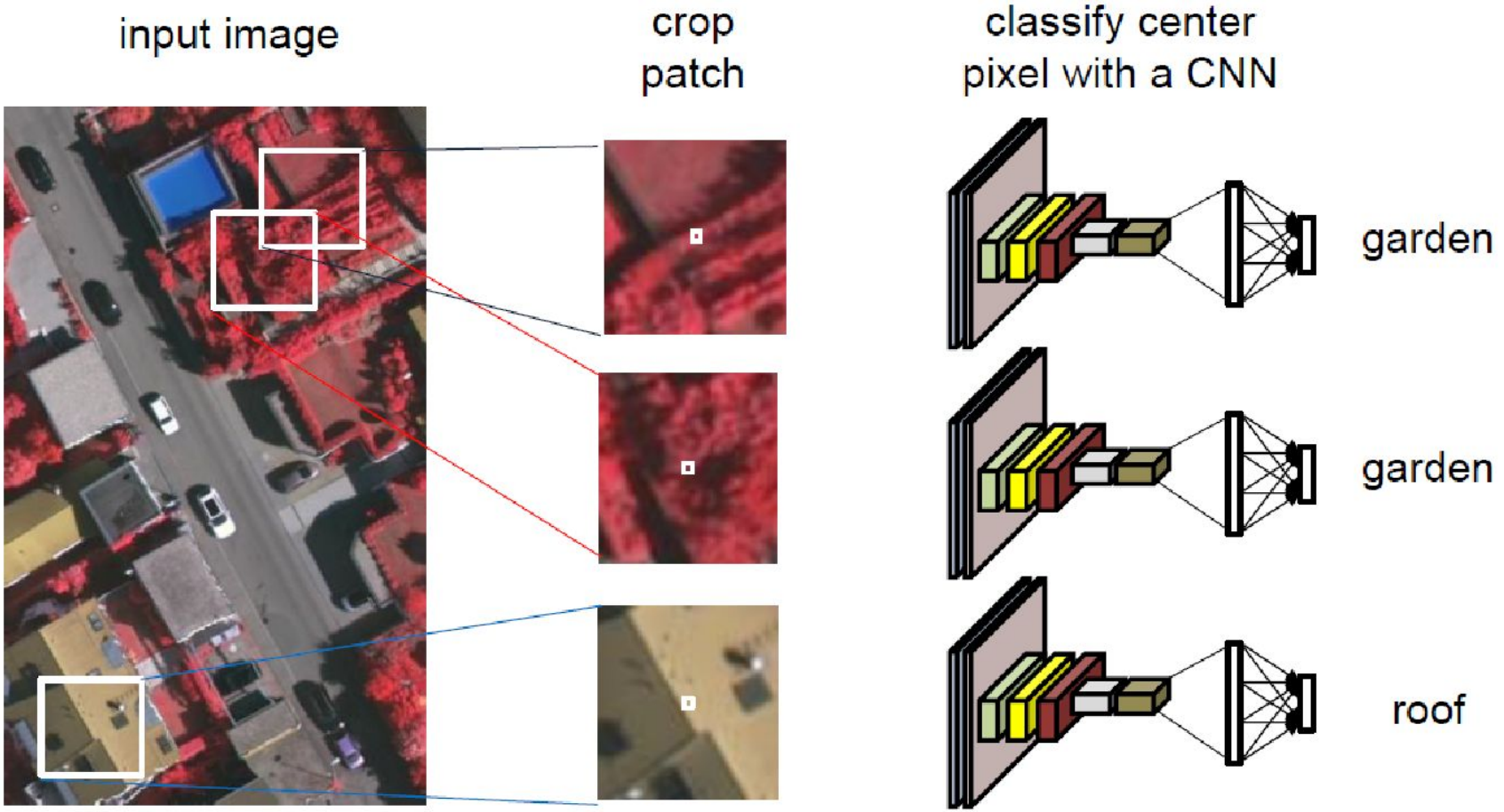
Imagem

ground-truth

Mapa de relevância



Nogueira et al. Learning to semantically segment high-resolution remote sensing images. ICPR 2016.

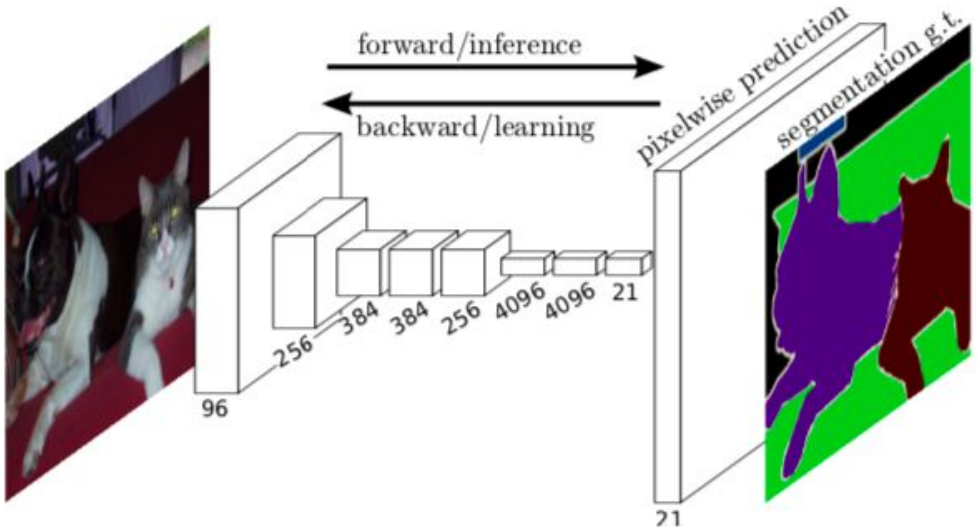


Operações redundantes devido aos overlaps. Muito ineficiente!

Fully Convolutional Networks (FCN)

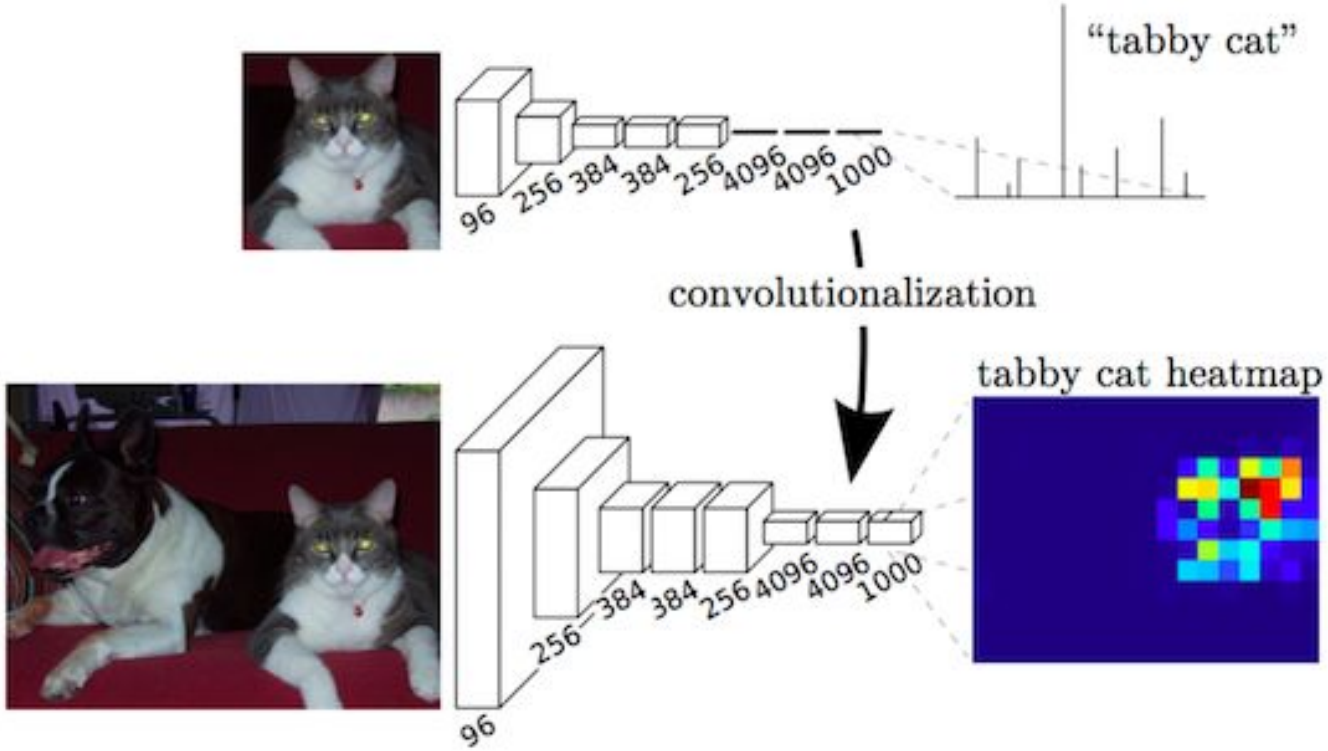
Fully Convolutional Networks

- Fully Convolutional Networks (FCNs)



J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. CVPR 2015.

Fully Convolutional Networks



J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. CVPR 2015.

Última camada da FCN

- A última camada deve ter a mesma resolução da imagem
- Todas as camadas que vimos até agora fazem downsampling da entrada, i.e. diminuem a resolução
- Como fazer o upscaling na última camada para retornar ao tamanho original? Como recuperar a localização que foi perdida durante as operações de pooling?
  Vamos retornar a esta pergunta em breve.
- Note que a penúltima camada contém um feature map com uma resolução muito menor que a da imagem original, e que deve ter um número de canais/kernels igual ao número de classes do dataset

Como fazer *upsampling in-network*?

Dada a seguinte entrada 2 x 2, como obter uma saída 4 x 4?

Entrada: 2x2

1	2
3	4

Como fazer *upsampling in-network*?

Dada a seguinte entrada 2 x 2, como obter uma saída 4 x 4?

Entrada: 2x2

1	2
3	4

Saída: 4x4

Nearest Neighbor

1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

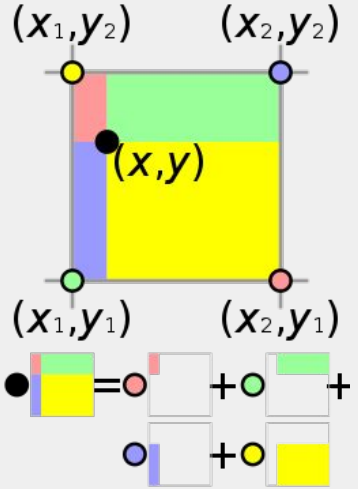
Bed of Nails
"Cama de pregos"

1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Interpolação bilinear

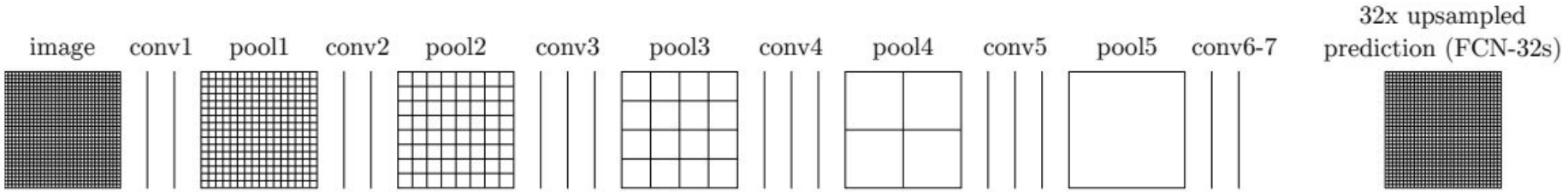
1		2	
3		4	

Interpolação bilinear.



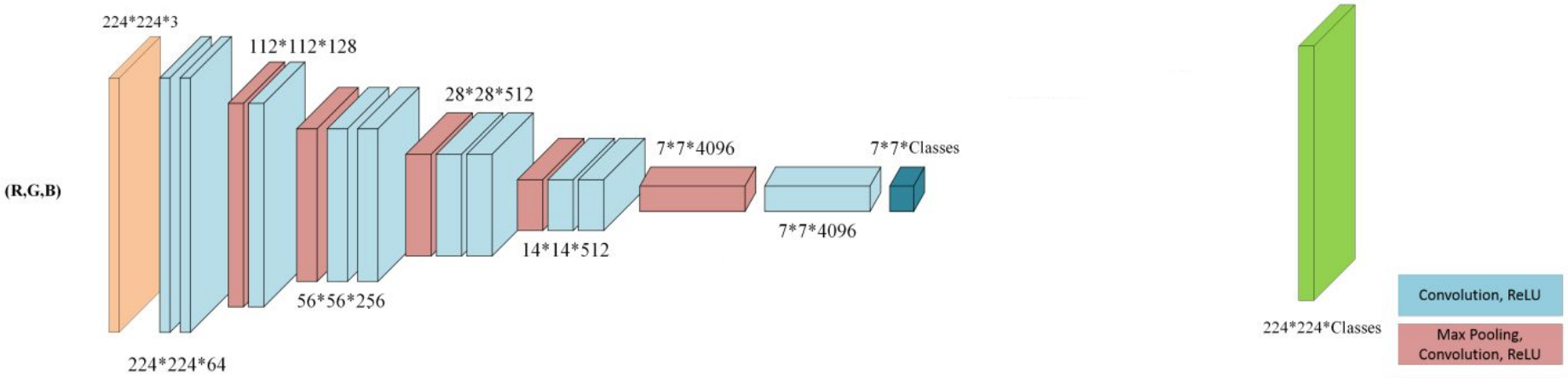
calcula cada saída y_{ij} a partir das 4 entradas mais próximas usando um mapa linear que depende apenas das posições relativas da saída e das entradas

Arquitetura da FCN-32s

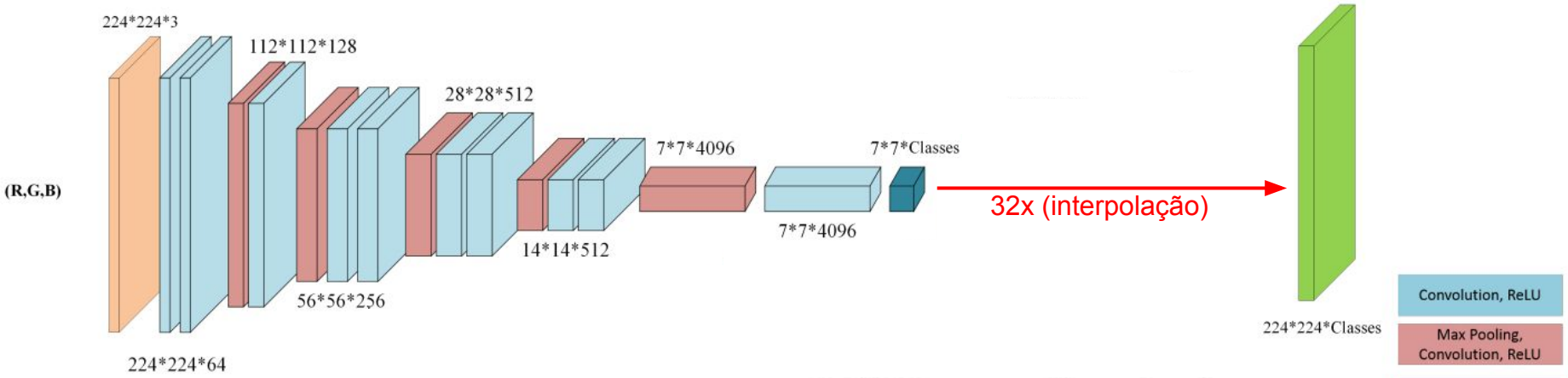


Long et al. Fully convolutional networks for semantic segmentation. In CVPR 2015.

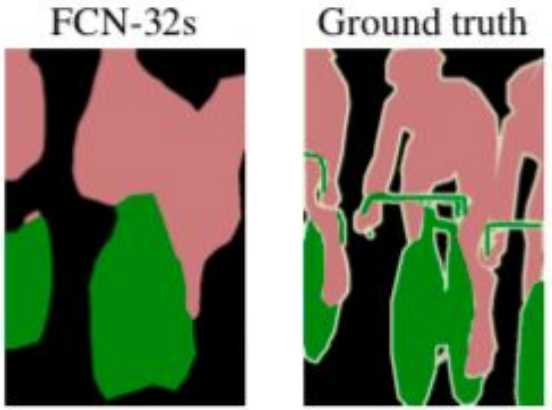
Arquitetura da FCN-32s



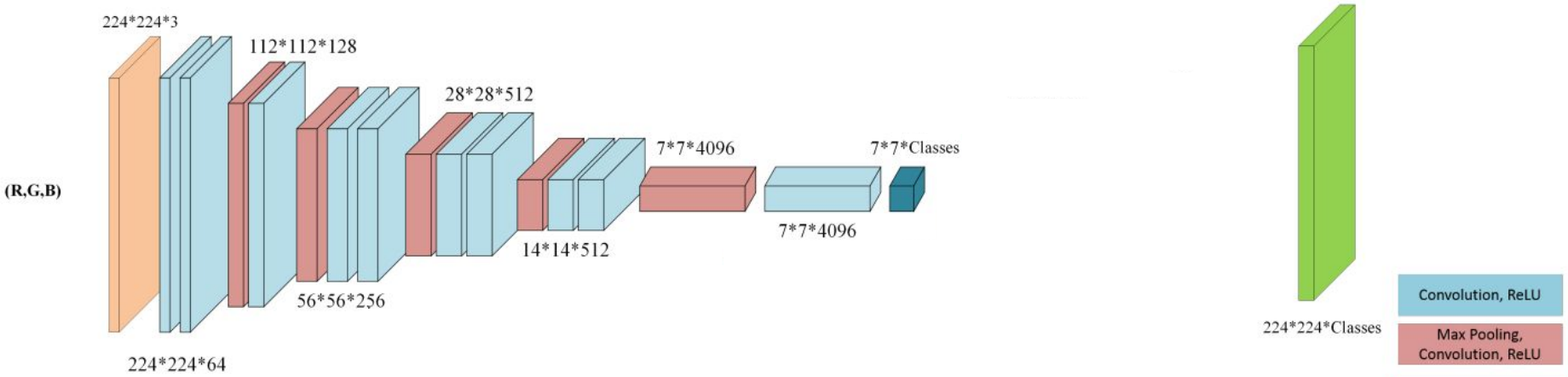
Arquitetura da FCN-32s



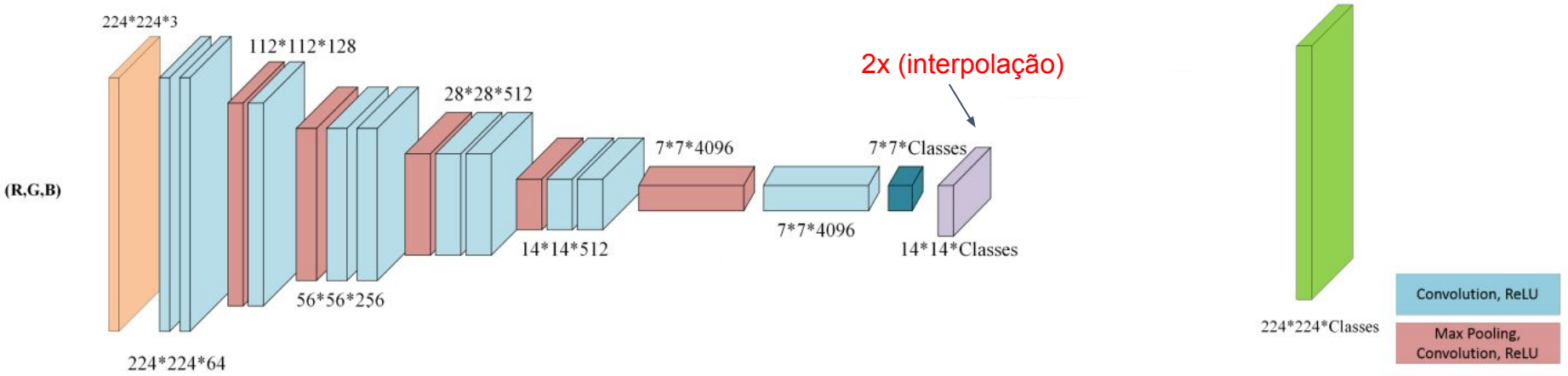
- FCN-32s: 32x conv7 (camada conv7 é upsampled 32x)
- Mesmo após o fine-tuning das redes pré-treinadas, os resultados são insatisfatoriamente grossos



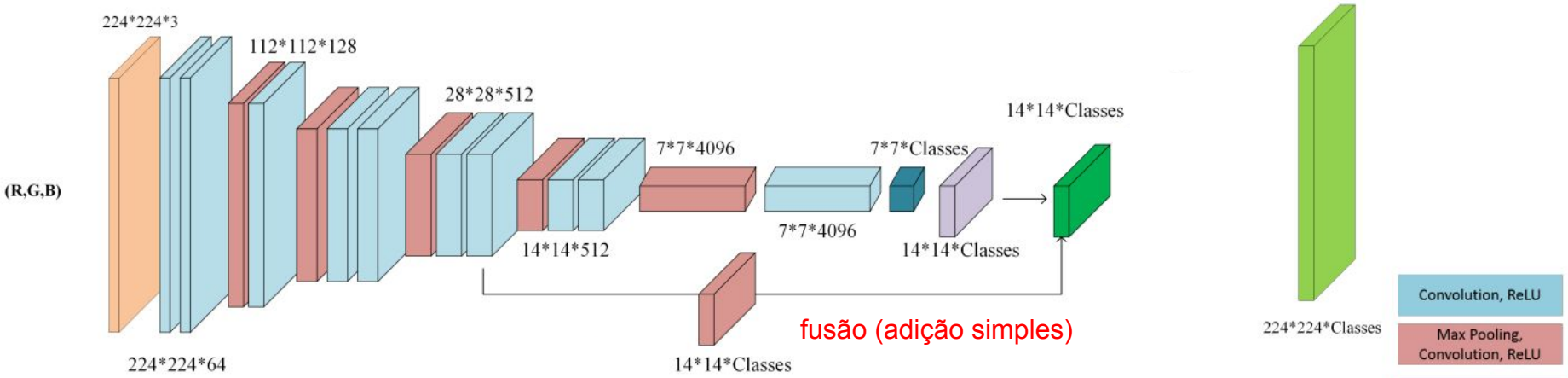
Arquitetura da FCN-16s



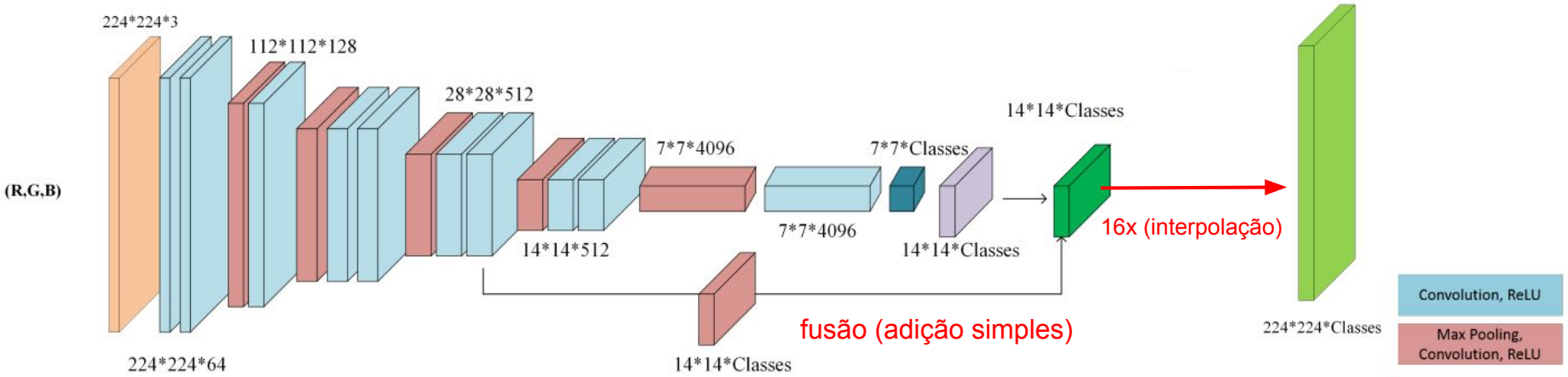
Arquitetura da FCN-16s



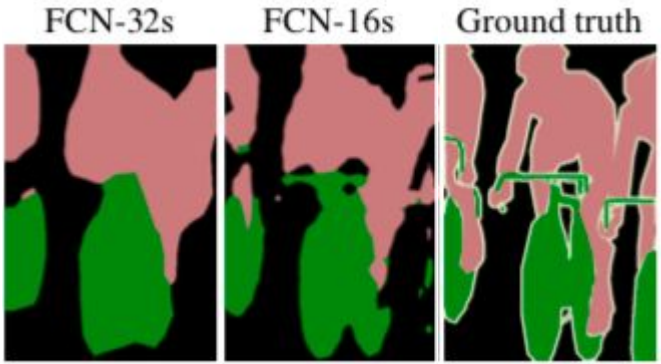
Arquitetura da FCN-16s



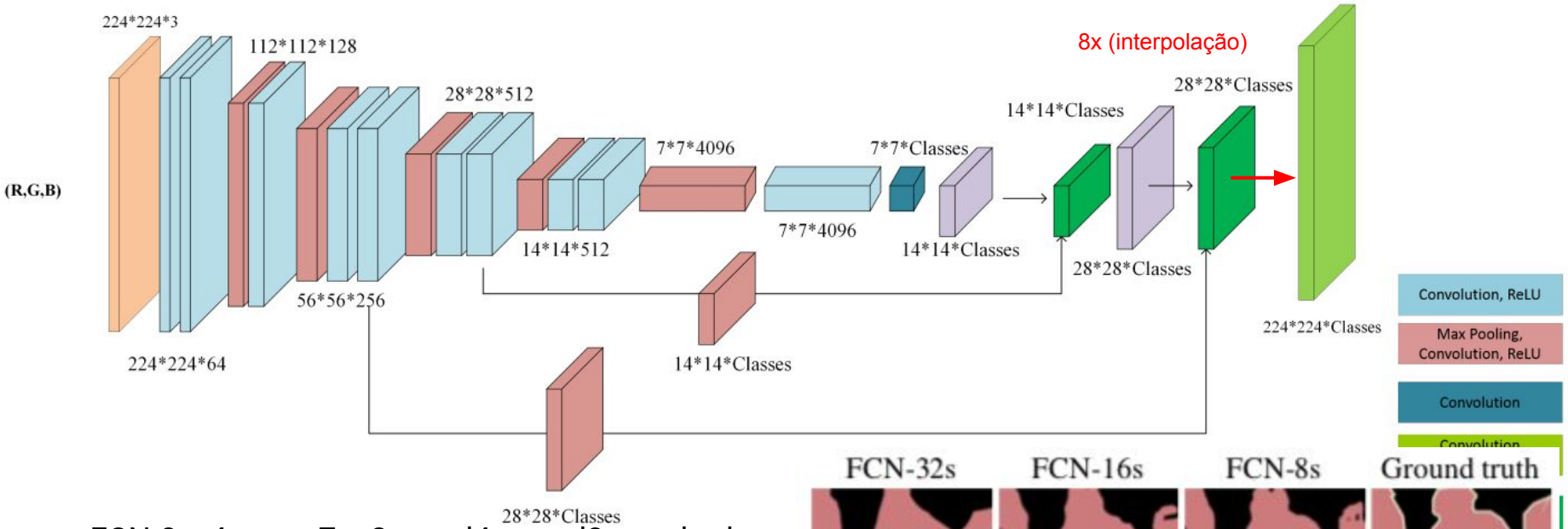
Arquitetura da FCN-16s

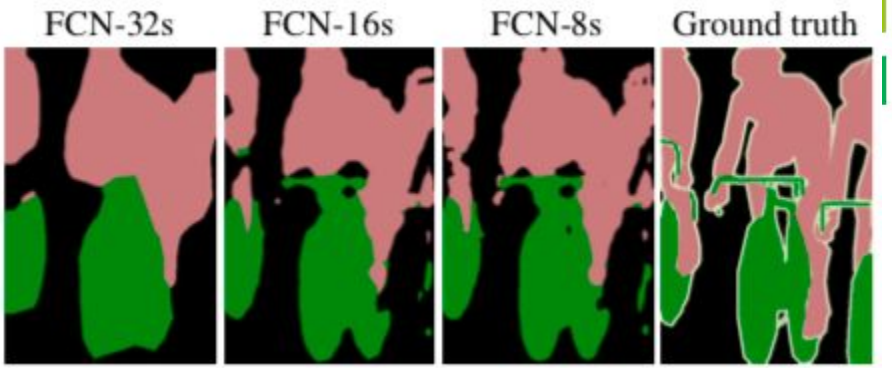


- Fusão de camadas intermediárias com as de predição pra complementar informação espacial do resultado final
- FCN-16s: 2x conv7 + pool4; resultado é upsampled 16x
- Diversas vantagens:
  - Fusão de informação de alto nível semântico com informação de baixo nível semântico



Arquitetura da FCN-8s





- FCN-8s: 4x conv7 + 2x pool4 + pool3; resultado é upsampled 8x
- Para recuperar informação espacial mais fina nas camadas finais do decoder, preciso ligar com camadas mais rasas do encoder (receptive field menor)

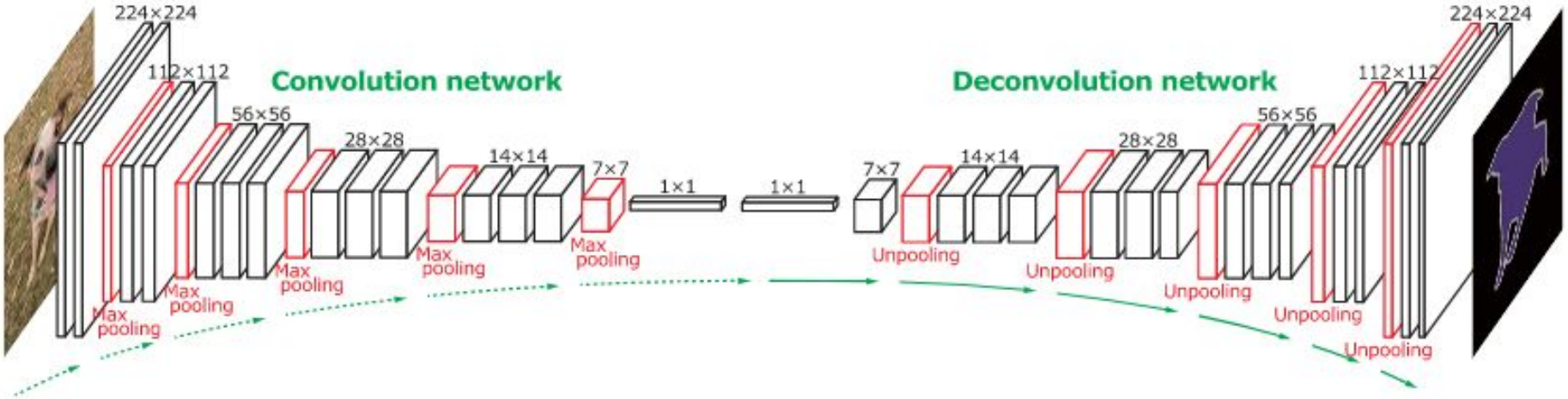
Contribuições principais

- Popularizou o uso de redes convolucionais end-to-end para segmentação semântica
- Adaptou redes pré-treinadas com o ImageNet para segmentação semântica
- Upsampling usando interpolação
- Introduziu “skip connections” para aumentar a granularidade do upsampling

Segmentação Semântica

Deconvolution Networks
(DeconvNets)

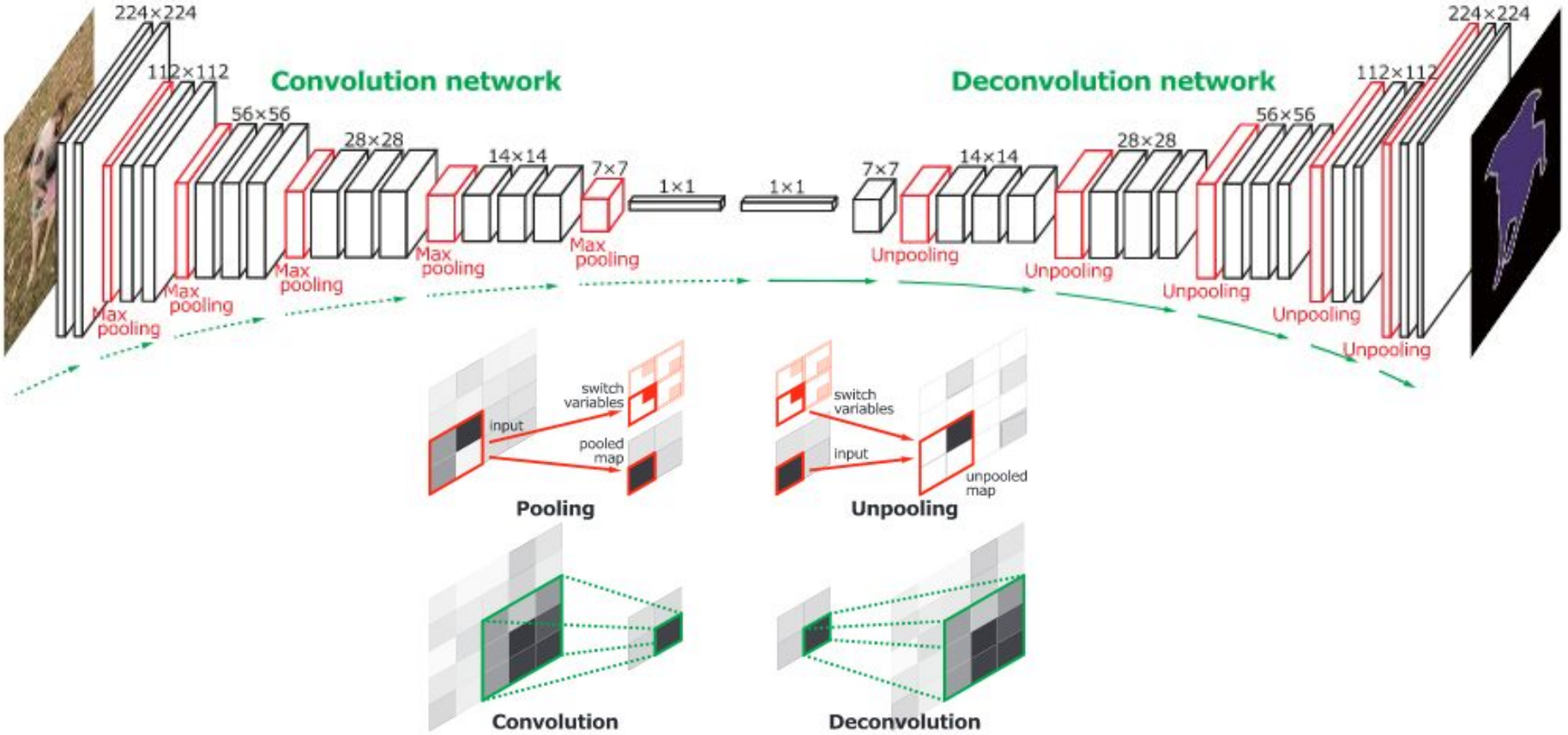
DeconvNets



Arquitetura “espelhada”!

Noh, H., Hong, S., & Han, B. (2015). Learning deconvolution network for semantic segmentation. In Proceedings of the IEEE international conference on computer vision (pp. 1520-1528).

DeconvNets


224×224
112×112
56×56
28×28
14×14
7×7
1×1
1×1
7×7
14×14
28×28
56×56
112×112
224×224
Convolution network
Deconvolution network
Max pooling
Max pooling
Max pooling
Max pooling
Max pooling
Unpooling
Unpooling
Unpooling
Unpooling
Unpooling
switch variables
input
pooled map
Pooling
switch variables
input
unpooled map
Unpooling
Convolution
Deconvolution

Como fazer *upsampling in-network*?

Dada a seguinte entrada 2 x 2, como obter uma saída 4 x 4?

Entrada: 2x2

1	2
3	4

Saída: 4x4

Nearest Neighbor

1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

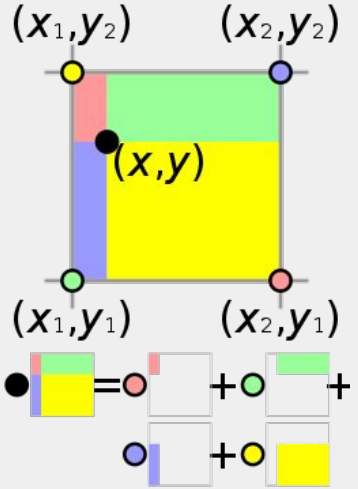
Bed of Nails
"Cama de pregos"

1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

Interpolação bilinear

1		2	
3		4	

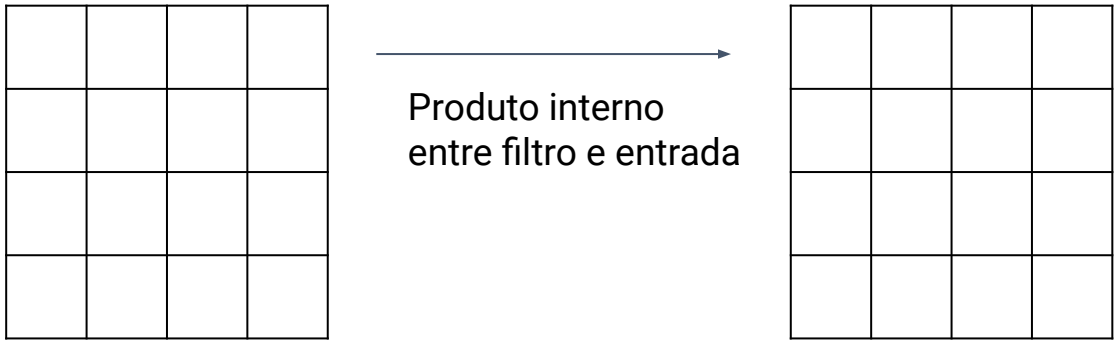
Interpolação bilinear.



calcula cada saída y_{ij} a partir das 4 entradas mais próximas usando um mapa linear que depende apenas das posições relativas da saída e das entradas

Upsampling com aprendizado: transpose convolution

Relembrando: convolução 3 x 3 típica, stride=1, pad=1

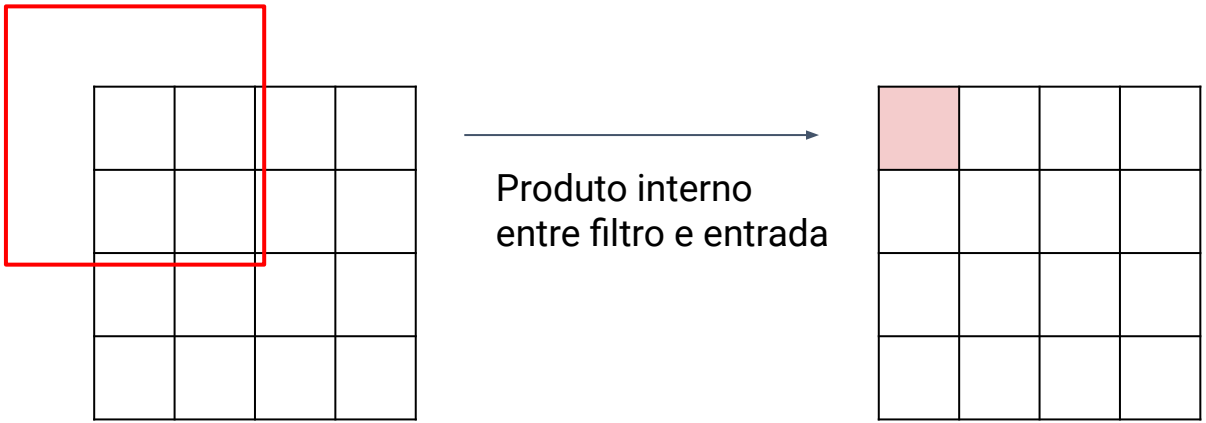


Entrada: 4x4

Saída: 4x4

Upsampling com aprendizado: transpose convolution

Relembrando: convolução 3 x 3 típica, stride=1, pad=1

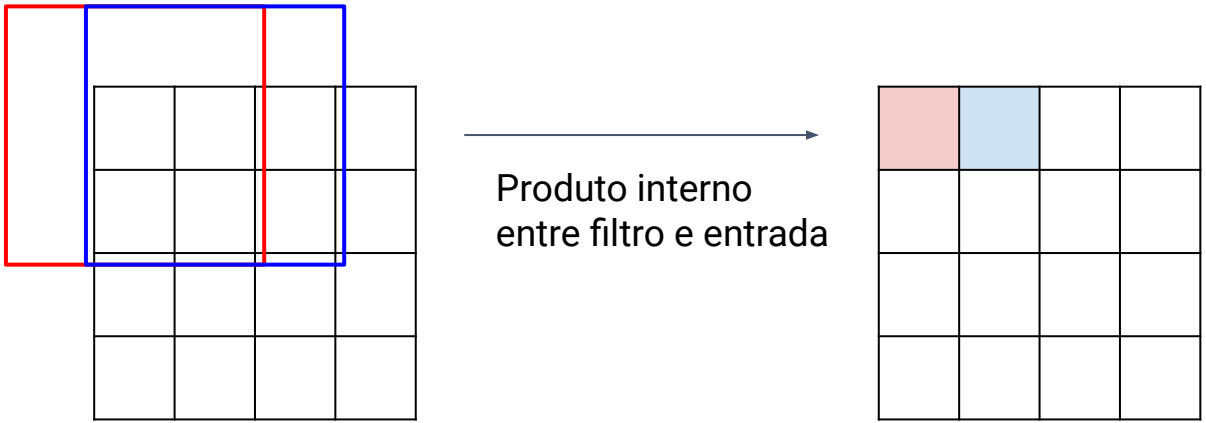


Entrada: 4x4

Saída: 4x4

Upsampling com aprendizado: transpose convolution

Relembrando: convolução 3 x 3 típica, stride=1, pad=1

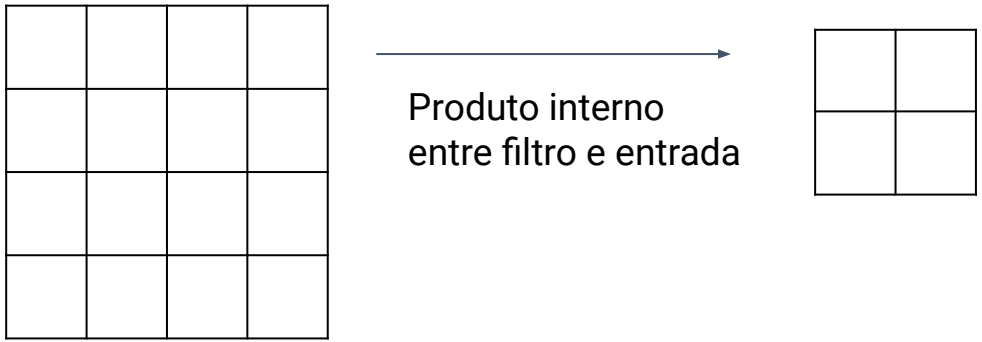


Entrada: 4x4

Saída: 4x4

Upsampling com aprendizado: transpose convolution

Relembrando: convolução 3 x 3 típica, stride=2, pad=1

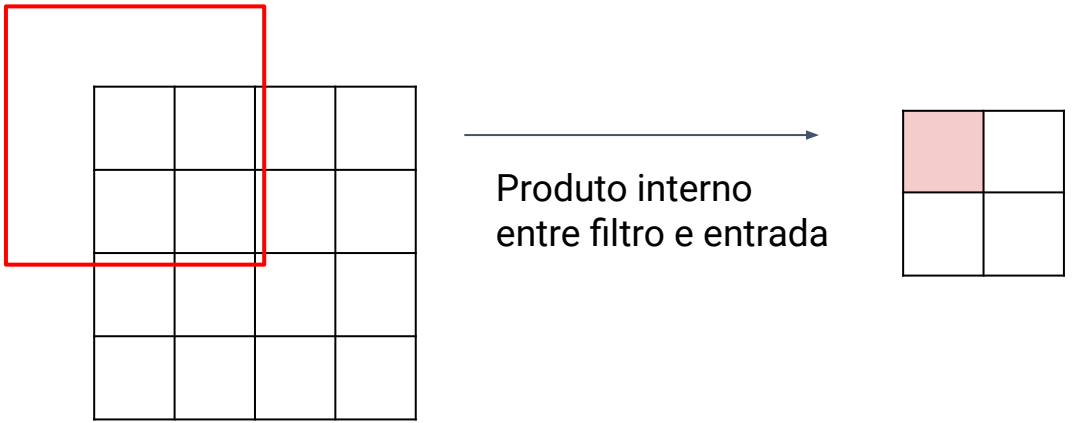


Entrada: 4x4

Saída: 2x2

Upsampling com aprendizado: transpose convolution

Relembrando: convolução 3 x 3 típica, stride=2, pad=1

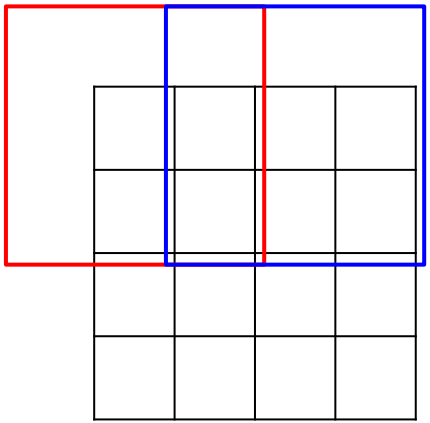


Entrada: 4x4

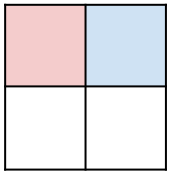
Saída: 2x2

Upsampling com aprendizado: transpose convolution

Relembrando: convolução 3 x 3 típica, stride=2, pad=1



Produto interno
entre filtro e entrada



Filtro move 2 pixels na entrada para cada pixel na saída

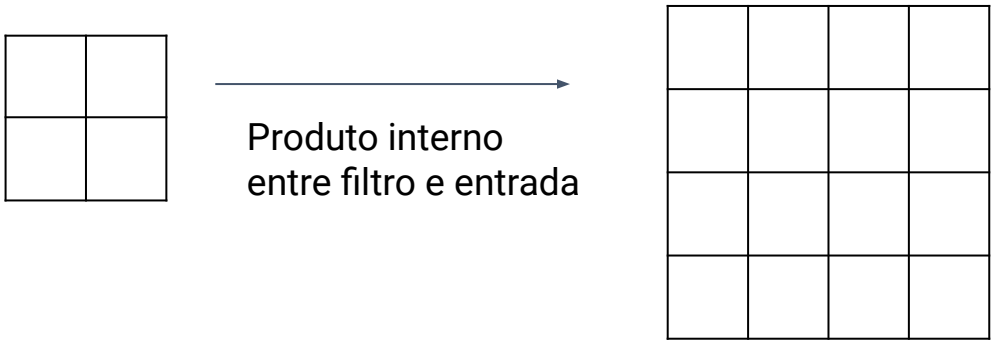
Stride dá a razão entre o deslocamento na entrada e na saída

Entrada: 4x4

Saída: 2x2

Upsampling com aprendizado: transpose convolution

convolução **transposta** 3 x 3, stride=2, pad=1

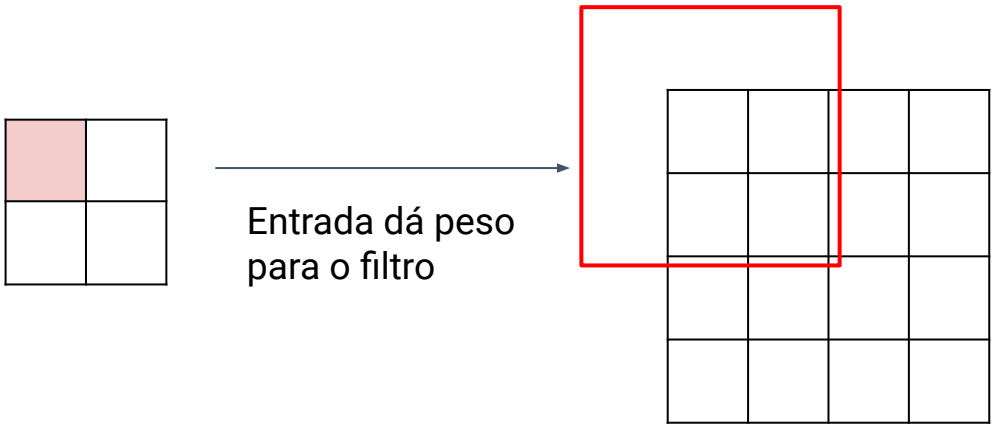


Entrada: 2x2

Saída: 4x4

Upsampling com aprendizado: transpose convolution

convolução **transposta** 3 x 3, stride=2, pad=1

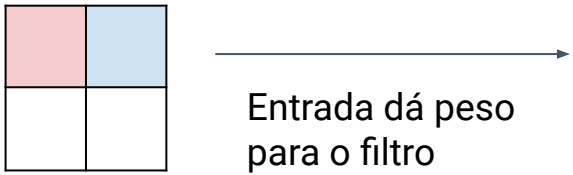


Entrada: 2x2

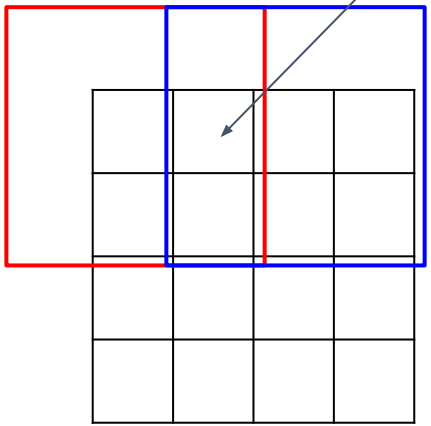
Saída: 4x4

Upsampling com aprendizado: transpose convolution

convolução **transposta** 3 x 3, stride=2, pad=1



Entrada dá peso para o filtro



Somamos onde saída se sobrepõe

Filtro move 2 pixels na saída para cada pixel na entrada

Stride dá a razão entre o movimento na saída e na entrada

Entrada: 2x2

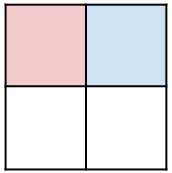
Saída: 4x4

Upsampling com aprendizado: transpose convolution

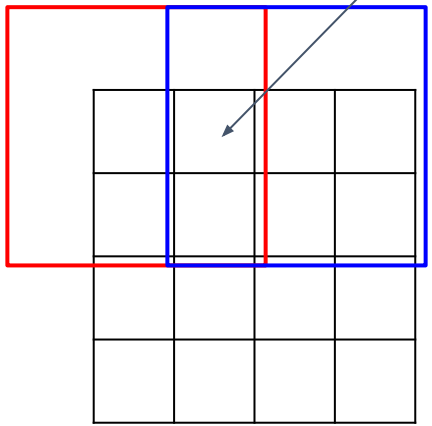
convolução **transposta** 3 x 3, stride=2, pad=1

Somamos onde saída se sobrepõe

Outros nomes:
- Deconvolution 👎
- Upconvolution
- Fractionally strided convolution
- Backward strided convolution



Entrada dá peso para o filtro



Filtro move 2 pixels na <u>saída</u> para cada pixel na <u>entrada</u>

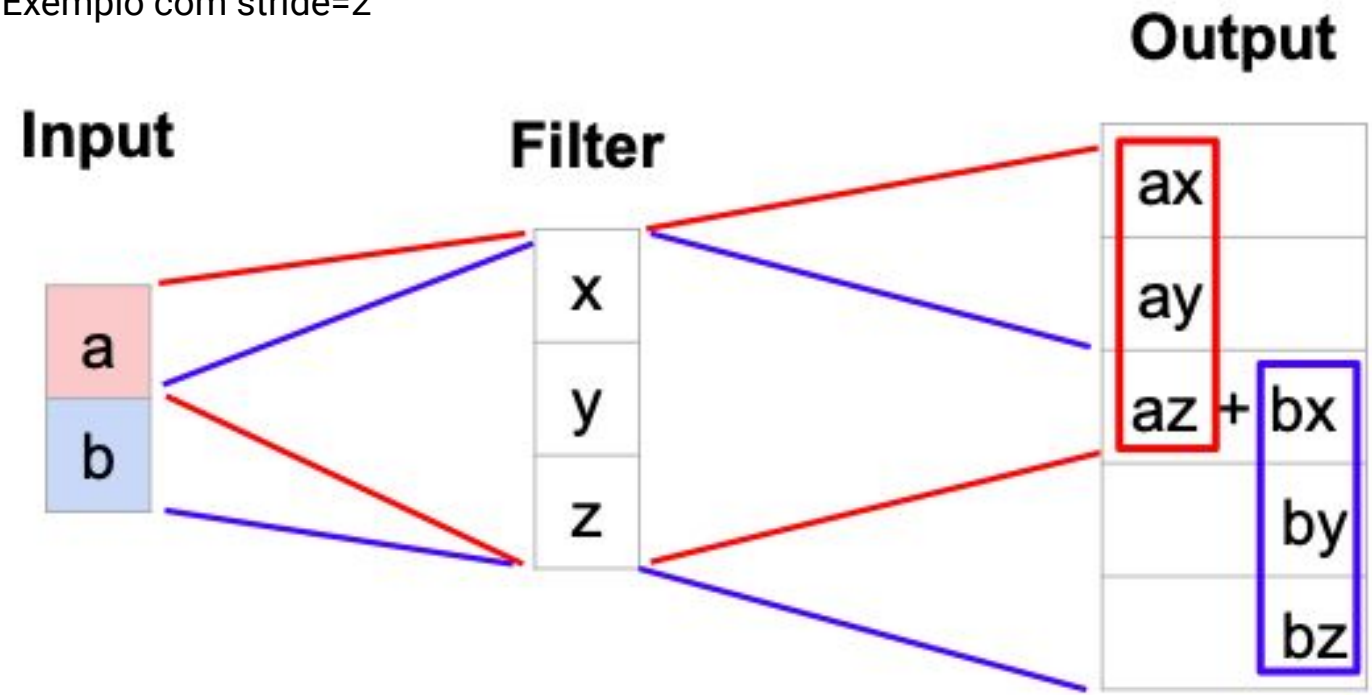
Stride dá a razão entre o movimento na saída e na entrada

Entrada: 2x2

Saída: 4x4

Convolução transposta: exemplo 1D

Exemplo com stride=2



Saída contém cópias do filtro ponderadas pela entrada, somando onde ocorre sobreposição da saída

O chão de o/i é 2,
o: tamanho da saída
i: tamanho da entrada

Convolução como multiplicação de matrizes (Exemplo 1D)

Seja **x** = (x,y,z) o vetor de pesos e **a** = (a,b,c,d) o vetor de entrada

Podemos expressar a convolução em termos de multiplicação de matriz

$$\mathbf{x} * \mathbf{a} = \mathbf{X}\mathbf{a}$$

$$\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & x & y & z & 0 & 0 \\ 0 & 0 & x & y & z & 0 \\ 0 & 0 & 0 & x & y & z \end{bmatrix} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \\ 0 \end{bmatrix} = \begin{bmatrix} ay + bz \\ ax + by + cz \\ bx + cy + dz \\ cx + dy \end{bmatrix}$$

Exemplo: 1D Conv, tamanho 3, stride=1, padding=1

Convolução como multiplicação de matrizes (Exemplo 1D)

Seja **x** = (x,y,z) o vetor de pesos e **a** = (a,b,c,d) o vetor de entrada

Podemos expressar a convolução em termos de multiplicação de matriz

$$\mathbf{x} * \mathbf{a} = \mathbf{X}\mathbf{a}$$

$$\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & x & y & z & 0 & 0 \\ 0 & 0 & x & y & z & 0 \\ 0 & 0 & 0 & x & y & z \end{bmatrix} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \\ 0 \end{bmatrix} = \begin{bmatrix} ay + bz \\ ax + by + cz \\ bx + cy + dz \\ cx + dy \end{bmatrix}$$

Exemplo: 1D Conv, tamanho 3, stride=1, padding=1

Convolução transposta multiplica pela transposta da mesma matriz

$$\mathbf{x} *^{\top} \mathbf{a} = \mathbf{X}^{\top}\mathbf{a}$$

$$\begin{bmatrix} x & 0 & 0 & 0 \\ y & x & 0 & 0 \\ z & y & x & 0 \\ 0 & z & y & x \\ 0 & 0 & z & y \\ 0 & 0 & 0 & z \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} ax \\ ay + bx \\ az + by + cx \\ bz + cy + dx \\ cz + dy \\ dz \end{bmatrix}$$

Quando stride=1, convolução transposta é uma convolução regular (com diferentes regras de padding)

Convolução como multiplicação de matrizes (Exemplo 1D)

Seja **x** = (x,y,z) o vetor de pesos e **a** = (a,b,c,d) o vetor de entrada

Podemos expressar a convolução em termos de multiplicação de matriz

$$\mathbf{x} * \mathbf{a} = \mathbf{X}\mathbf{a}$$

$$\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & 0 & x & y & z & 0 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \\ 0 \end{bmatrix} = \begin{bmatrix} ay + bz \\ bx + cy + dz \end{bmatrix}$$

Exemplo: 1D Conv, tamanho 3, stride=2, padding=1

Convolução como multiplicação de matrizes (Exemplo 1D)

Seja **x** = (x,y,z) o vetor de pesos e **a** = (a,b,c,d) o vetor de entrada

Podemos expressar a convolução em termos de multiplicação de matriz

$$\mathbf{x} * \mathbf{a} = \mathbf{X}\mathbf{a}$$

$$\begin{bmatrix} x & y & z & 0 & 0 & 0 \\ 0 & 0 & x & y & z & 0 \end{bmatrix} \begin{bmatrix} 0 \\ a \\ b \\ c \\ d \\ 0 \end{bmatrix} = \begin{bmatrix} ay + bz \\ bx + cy + dz \end{bmatrix}$$

Exemplo: 1D Conv, tamanho 3, stride=2, padding=1

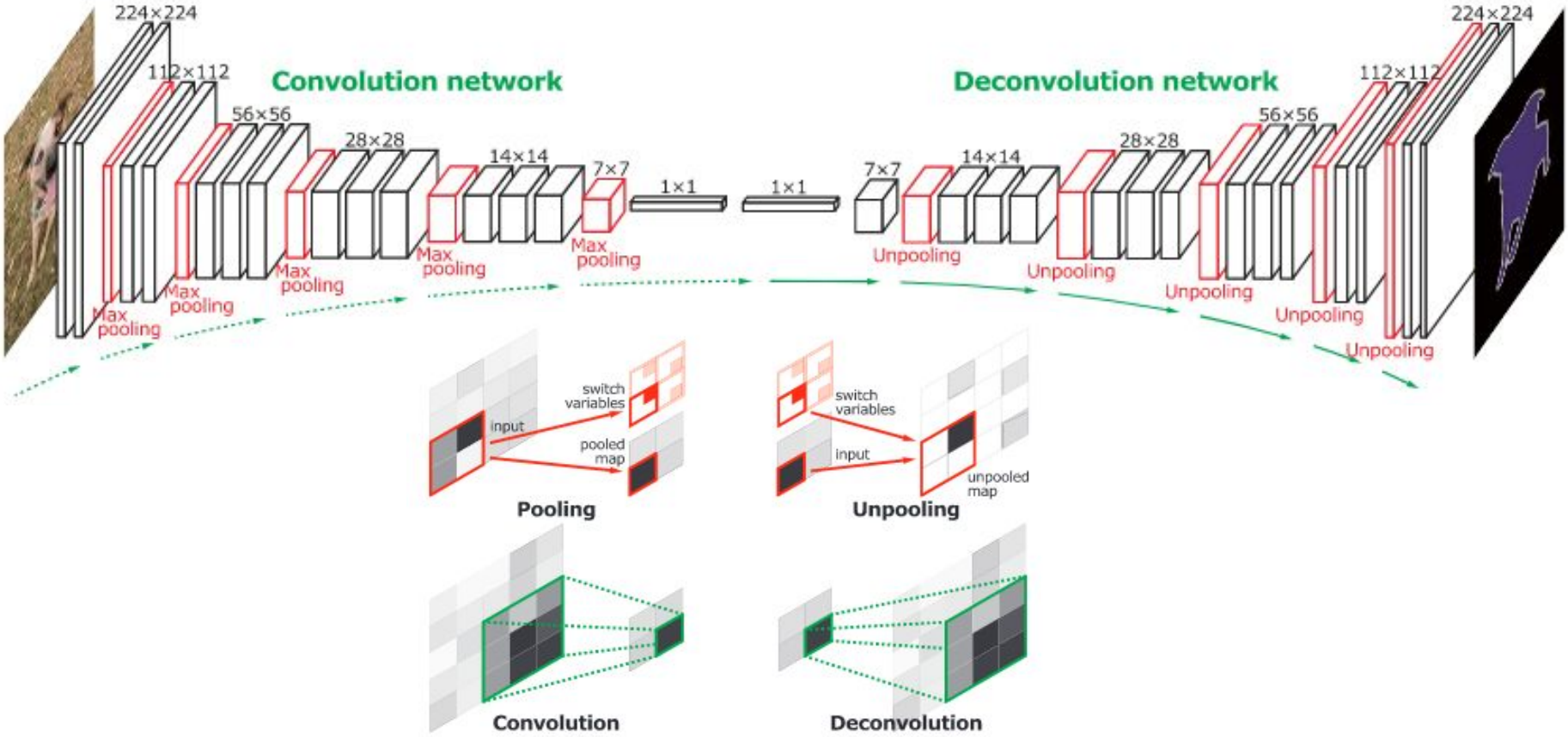
Convolução transposta multiplica pela transposta da mesma matriz

$$\mathbf{x} *^{\top} \mathbf{a} = \mathbf{X}^{\top}\mathbf{a}$$

$$\begin{bmatrix} x & 0 \\ y & 0 \\ z & x \\ 0 & y \\ 0 & z \\ 0 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} ax \\ ay \\ az + bx \\ by \\ bz \\ 0 \end{bmatrix}$$

Quando stride >1, convolução transposta não é mais uma convolução normal

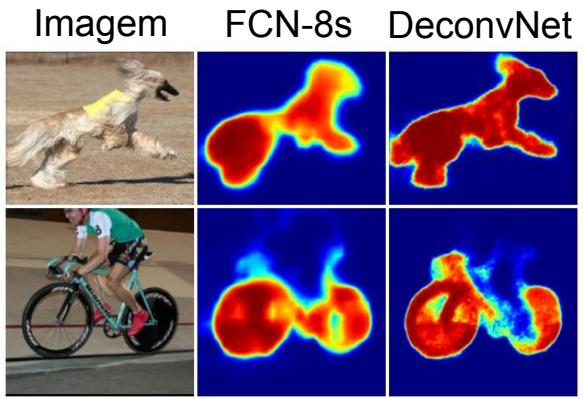
DeconvNets

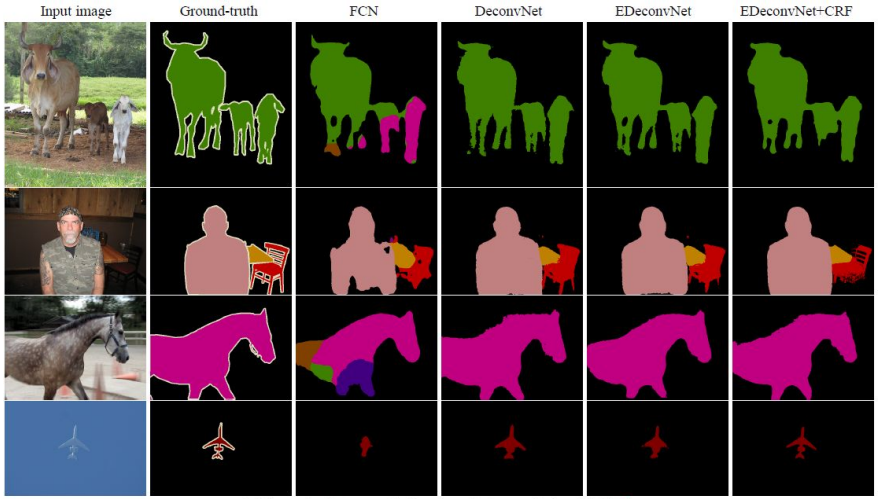

224×224
112×112
56×56
28×28
14×14
7×7
1×1
1×1
7×7
14×14
28×28
56×56
112×112
224×224
Convolution network
Deconvolution network
Max pooling
Max pooling
Max pooling
Max pooling
Max pooling
Unpooling
Unpooling
Unpooling
Unpooling
Unpooling
switch variables
input
pooled map
Pooling
switch variables
input
unpooled map
Unpooling
Convolution
Deconvolution

DeconvNets

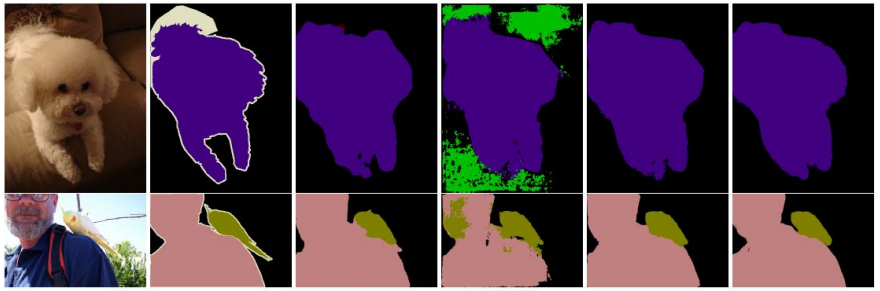
Problemas:

- Dificuldade de convergência
- Muitos parâmetros

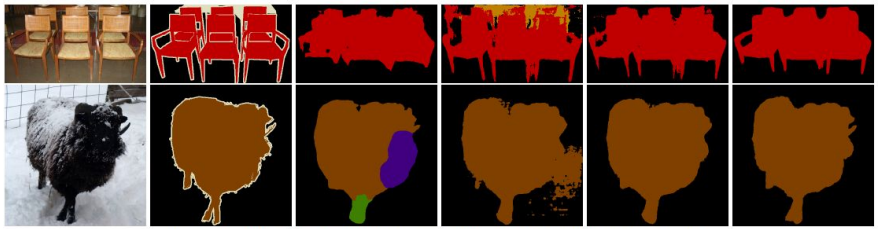




(a) Examples that our method produces better results than FCN [17].



(b) Examples that FCN produces better results than our method.



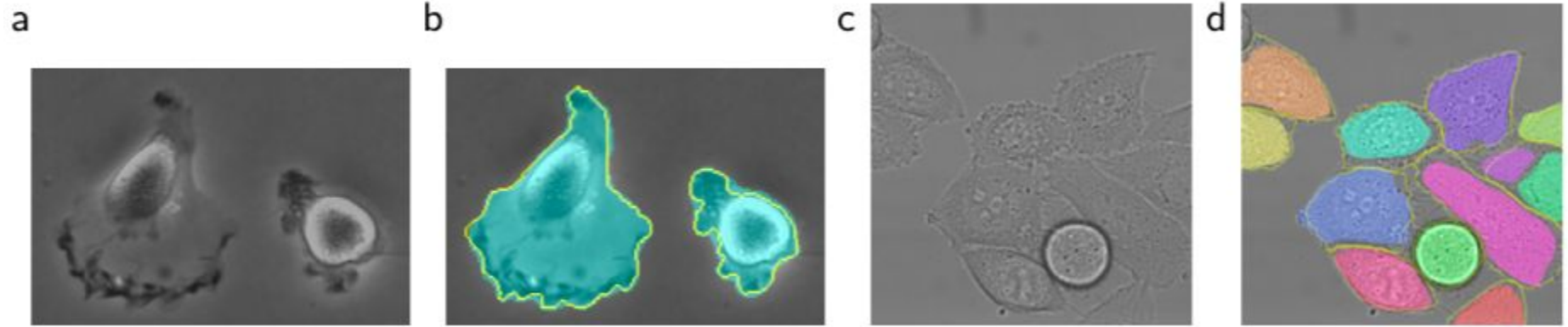
(c) Examples that inaccurate predictions from our method and FCN are improved by ensemble.

Segmentação Semântica

U-Nets

U-Net

- Proposto no contexto de processamento de imagens biomédicas
  - ISBI challenge for segmentation of neuronal structures
  - Necessidade de se fazer classificação densa (por pixel)
  - Poucos exemplos de treinamento (~30 por aplicação)
  - Células que se tocam; espaço muito pequeno entre objetos
- Solução reutilizável em diferentes domínios



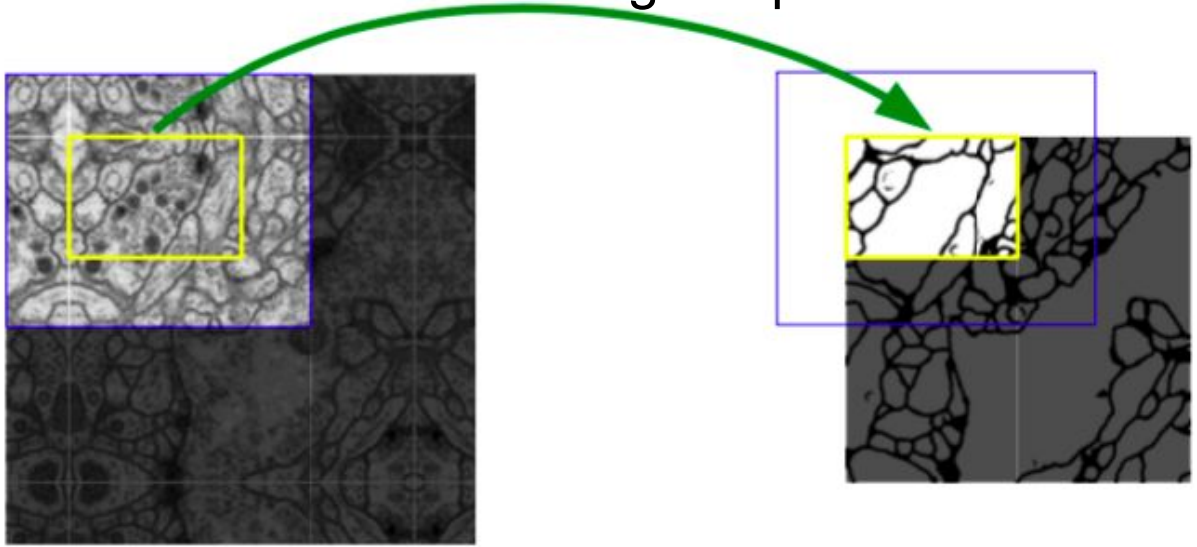
PhC-U373: 35 imagens

PhC-U373: 20 imagens

Ronneberger et al. U-net: Convolutional networks for biomedical image segmentation. MICCAI 2015.

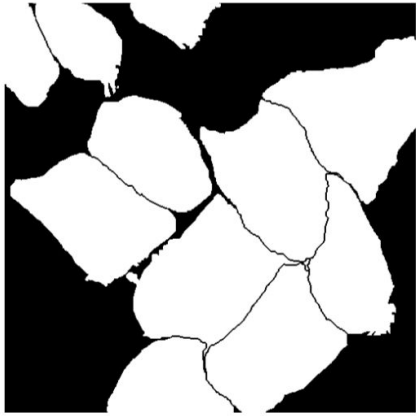
U-Net: principais ideias

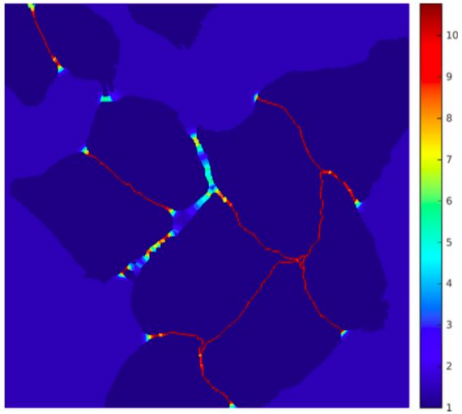
- Arquitetura encoder-decoder
- Augumentação de imagens
- Padding: em vez de completar com 0's, espelhar a imagem nas bordas. Técnica utilizada na imagem que entra na rede.



U-Net: principais ideias

- Função de perda que dá pesos aos diferentes pixels
  - Balancear as diferentes frequências das classes (nas imagens há mais exemplos de uma classe que de outras)
  - Forçar a rede a aprender pequenas bordas de separação presentes entre células que se tocam





$$w(\mathbf{x}) = w_c(\mathbf{x}) + w_0 \cdot \exp\left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2}\right)$$

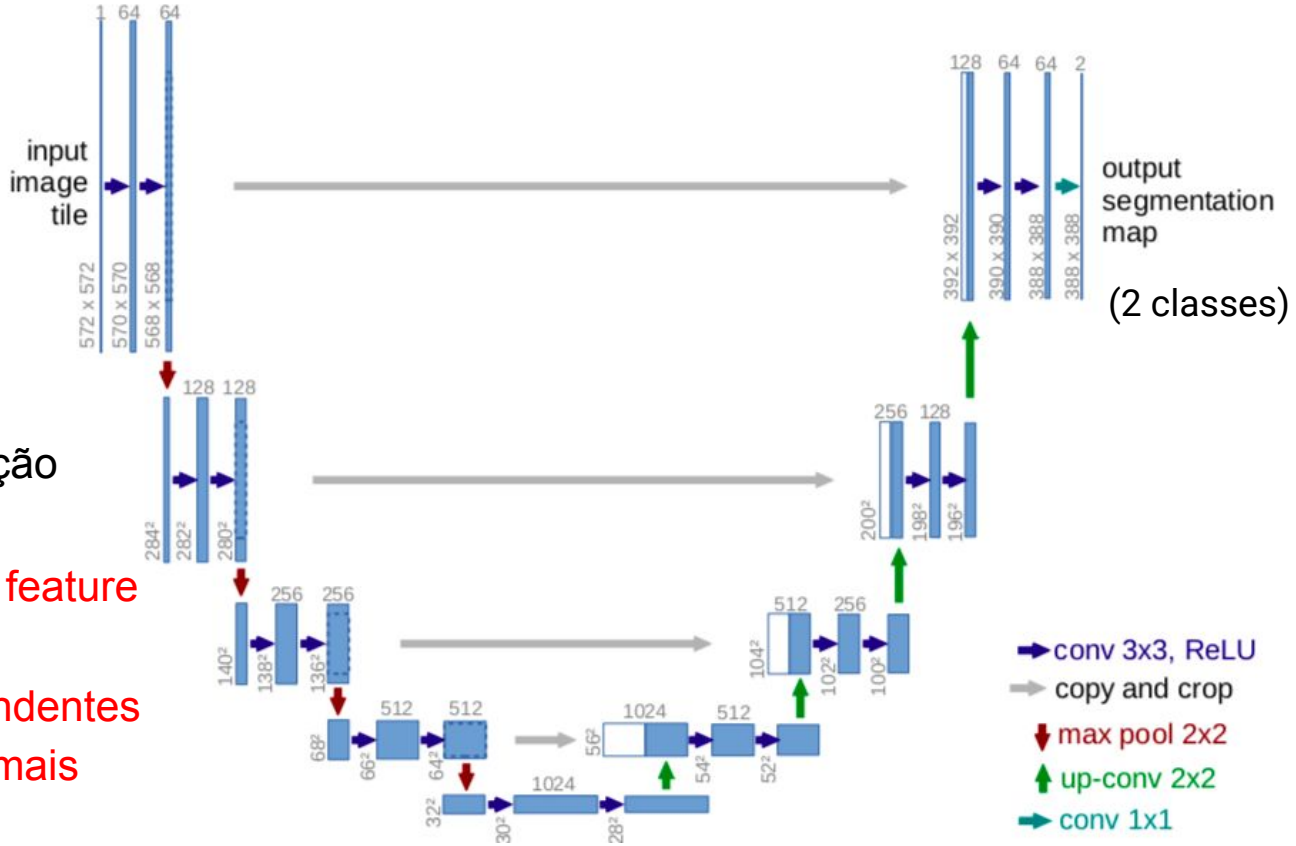
compensação de frequências das classes

Distância até borda mais próxima

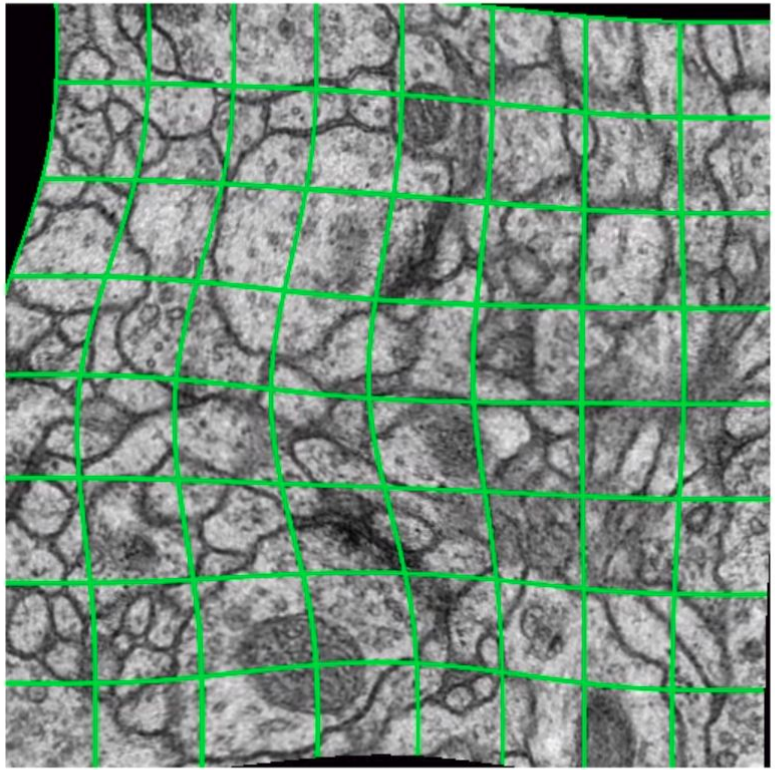
Distância até 2a. borda mais próxima

U-Net: principais ideias

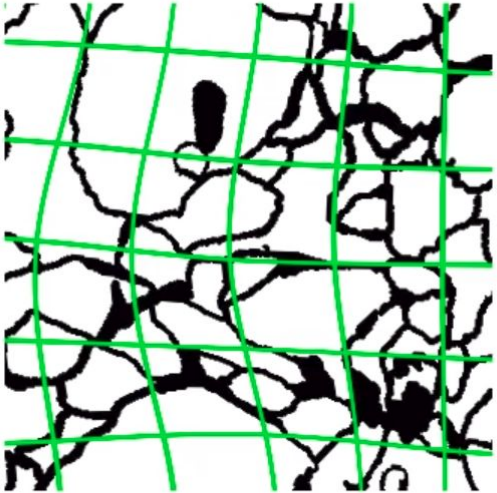
- Camadas dedicadas à recuperação da informação espacial
- No decoder: concatenar feature maps das camadas de downsampling correspondentes para obter localizações mais precisas



Augment Training Data using Deformations



resulting deformed image
(for visualization: no rotation, no shift, no extrapolation)



correspondingly deformed
manual labels

Contribuições principais da U-Net

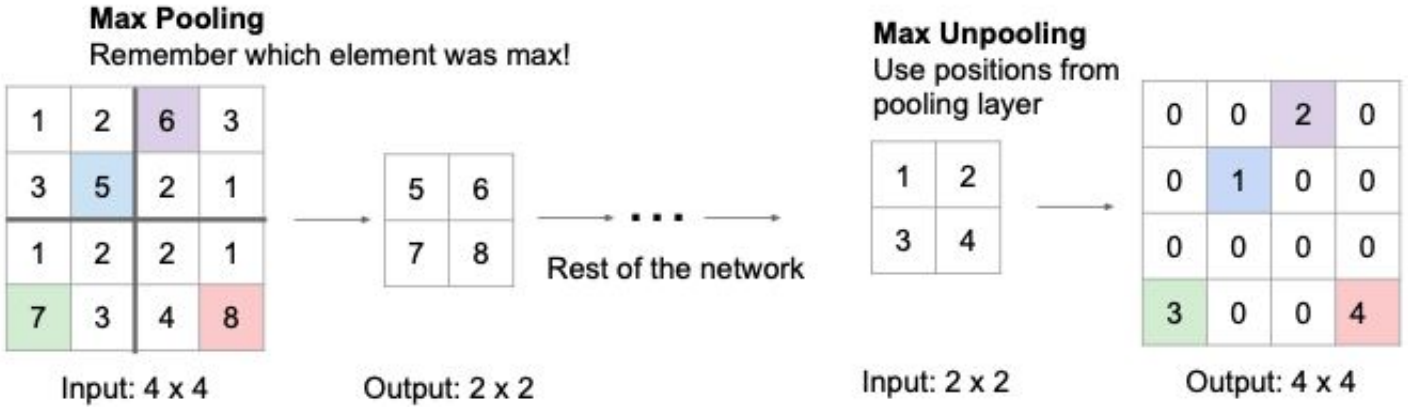
- Camadas dedicadas à recuperação da informação espacial
- No decoder: concatenar feature maps das camadas de downsampling correspondentes para obter localizações mais precisas
- Aumento de dados para imagens microscópicas
- Função de custo que permite aprender pequenas bordas de separação entre objetos

Segmentação Semântica

SegNet

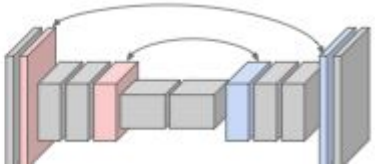
Como fazer upsampling in-network (unpooling)?

Quando fazemos max pooling, perdemos a informação de que elemento da camada foi utilizado.
Podemos guardar estas posições e utilizá-las posteriormente:



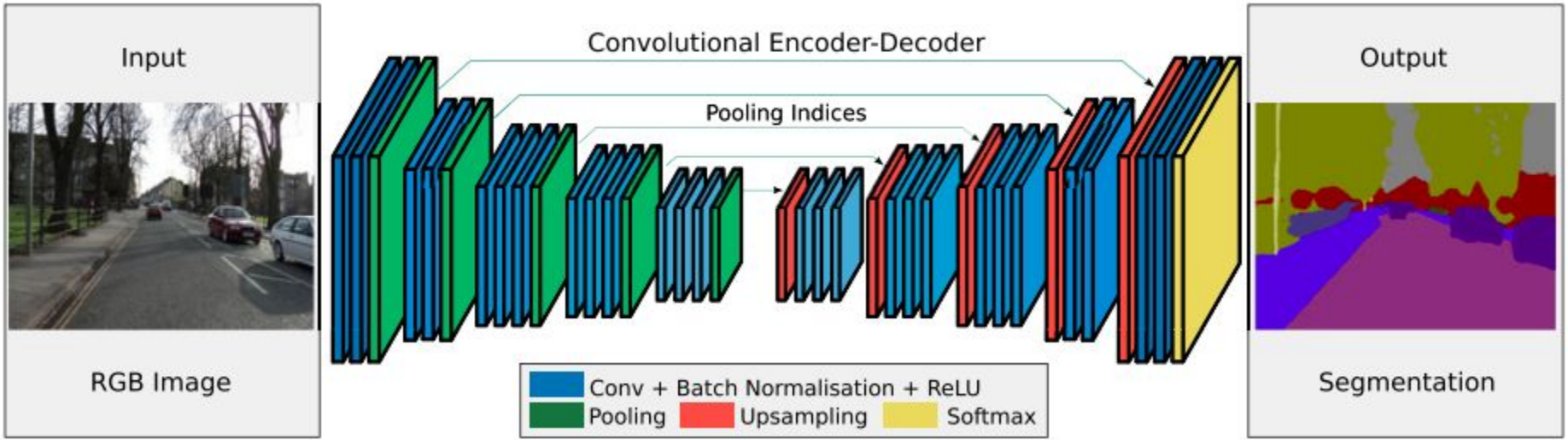
Semelhante à cama de pregos, mas coloca os elementos correspondentes nas posições originais

Corresponding pairs of downsampling and upsampling layers



SegNet

- Arquiteturas Encoder-Decoder
- Uso de índices de pooling do Encoder para a reconstrução do Decoder



Badrinarayanan et al. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation." IEEE TPAMI 2016.

SegNet: resultados no CamVid dataset

Method	Building	Tree	Sky	Car	Sign-Symbol	Road	Pedestrian	Fence	Column-Pole	Side-walk	Bicyclist	Class avg.	Global avg.
SfM+Appearance [28]	46.2	61.9	89.7	68.6	42.9	89.5	53.6	46.6	0.7	60.5	22.5	53.0	69.1
Boosting [29]	61.9	67.3	91.1	71.1	58.5	92.9	49.5	37.6	25.8	77.8	24.7	59.8	76.4
Dense Depth Maps [32]	85.3	57.3	95.4	69.2	46.5	**98.5**	23.8	44.3	22.0	38.1	28.7	55.4	82.1
Structured Random Forests [31]	n/a											51.4	72.5
Neural Decision Forests [64]	n/a											56.1	82.1
Local Label Descriptors [65]	80.7	61.5	88.8	16.4	n/a	98.0	1.09	0.05	4.13	12.4	0.07	36.3	73.6
Super Parsing [33]	87.0	67.1	96.9	62.7	30.1	95.9	14.7	17.9	1.7	70.0	19.4	51.2	83.3
SegNet (3.5K dataset training - 140K)	**89.6**	**83.4**	96.1	**87.7**	52.7	96.4	**62.2**	**53.45**	**32.1**	**93.3**	**36.5**	**71.20**	**90.40**
CRF based approaches													
Boosting + pairwise CRF [29]	70.7	70.8	94.7	74.4	55.9	94.1	45.7	37.2	13.0	79.3	23.1	59.9	79.8
Boosting+Higher order [29]	84.5	72.6	**97.5**	72.7	34.1	95.3	34.2	45.7	8.1	77.6	28.5	59.2	83.8
Boosting+Detectors+CRF [30]	81.5	76.6	96.2	78.7	40.2	93.9	43.0	47.6	14.3	81.5	33.9	62.5	83.8

Contribuições principais da SegNet

- Índices do maxpooling transferidos para o decoder para aumentar a resolução da segmentação
    - Como as features do encoder não são copiadas (como na FCN), SegNet é mais eficiente no uso de memória

Network	Forward pass(ms)	Backward pass(ms)	GPU training memory (MB)	GPU inference memory (MB)	Model size (MB)
SegNet	422.50	488.71	6803	**1052**	117
DeepLab-LargeFOV [3]	**110.06**	**160.73**	**5618**	1993	83
FCN (learnt deconv) [2]	317.09	484.11	9735	1806	539
DeconvNet [4]	474.65	602.15	9731	1872	877

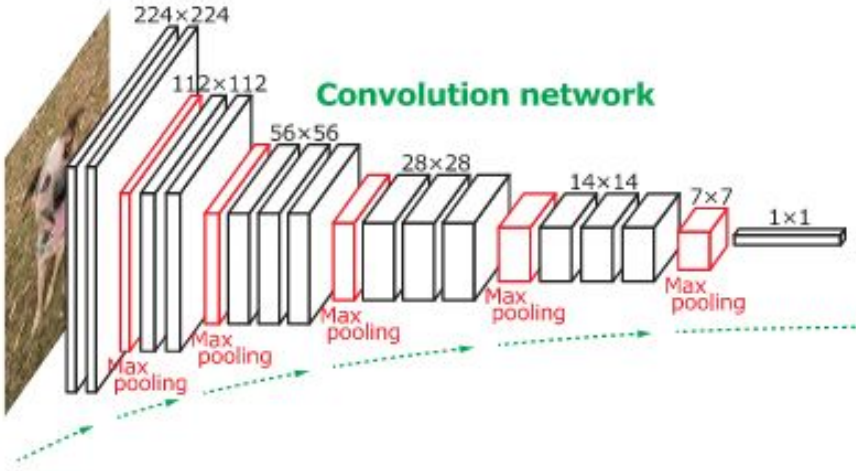
Segmentação Semântica

Convolução Dilatada
(a.k.a. *Atrous Convolution*)

Segmentação e CNNs

Problema:

- **Down-sampling** causa perda de informação informação espacial.

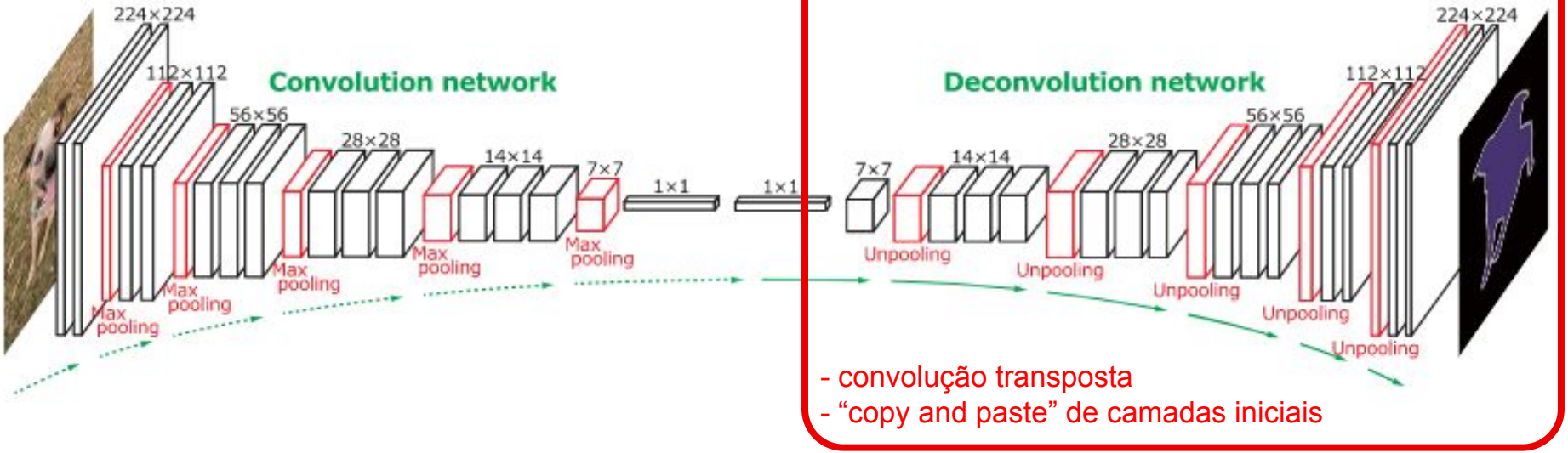


Segmentação e CNNs

Problema:

- **Down-sampling** causa perda de informação informação espacial.

Possível solução



Segmentação e CNNs

Problema:

- x **Down-sampling** causa perda de informação informação espacial.

Outra solução:

- Convolução dilatada (‘Holes’ algorithm).

Convolução Dilatada

Introduz um parâmetro para as camadas convolucionais: a taxa de dilatação.

- Convolução dilatada para o sinal 1-D:

$$y[i] = \sum_{k=1}^{K} x[i + r \cdot k] w[k]$$

$x[i]$ 1-D input signal

$w[k]$ filter of length K

r *rate* parameter corresponds to the stride with which we sample the input signal.

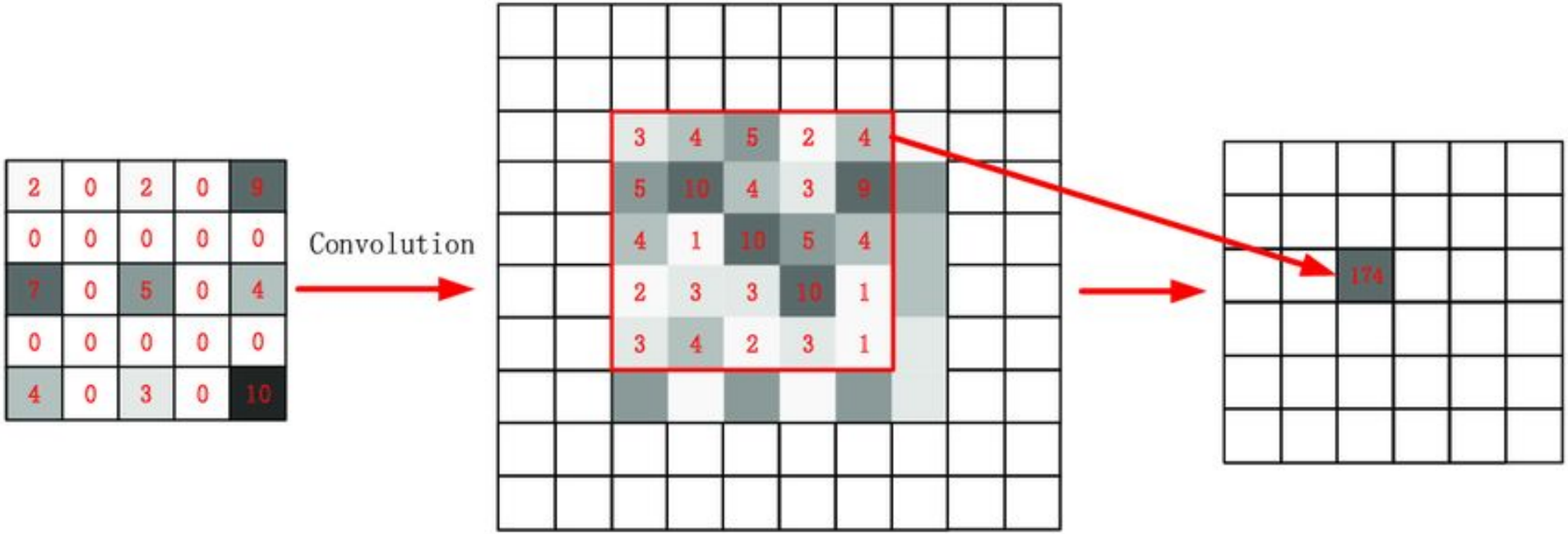
$y[i]$ output of atrous convolution.

Introduz zeros entre os valores do filtro

Nota: convolução padrão é um caso especial em que a taxa é *r = 1*.

Convolução Dilatada

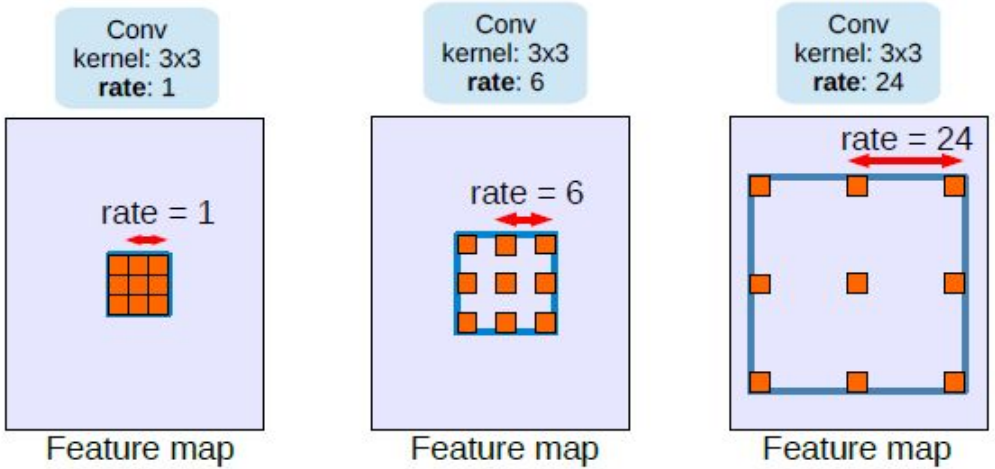
Introduz um parâmetro para as camadas convolucionais: a taxa de dilatação.



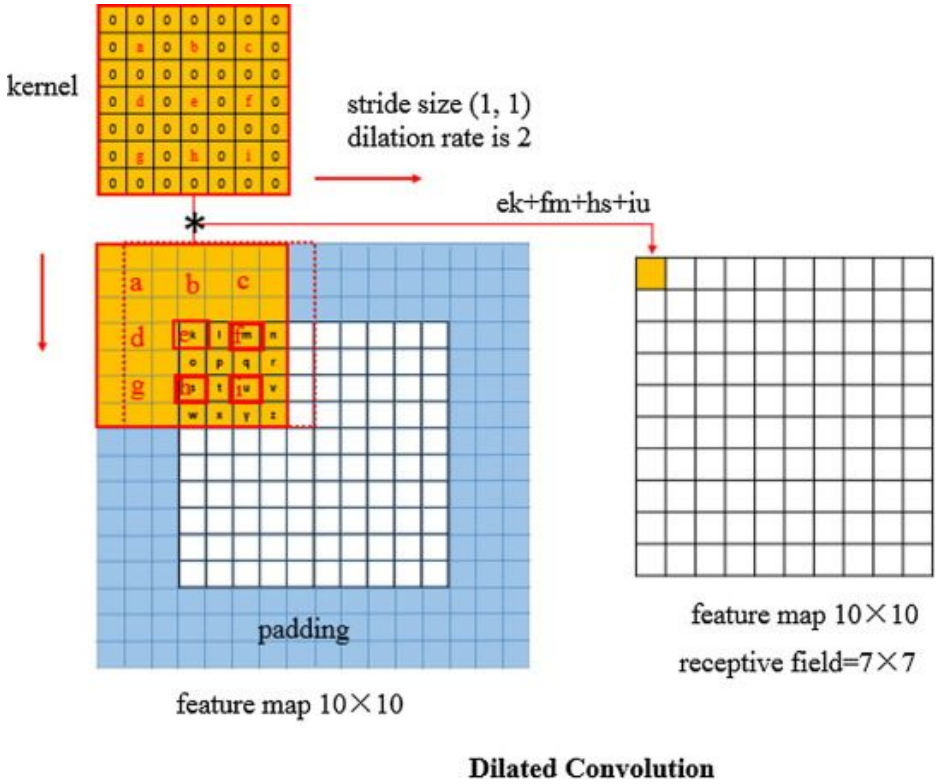
Convolução Dilatada

Introduz um parâmetro para as camadas convolucionais: a taxa de dilatação.

- Efeito:
  - um kernel 3x3 com uma taxa de dilatação de 2 terá o mesmo campo de visão que um kernel 5x5, enquanto usa apenas 9 parâmetros.



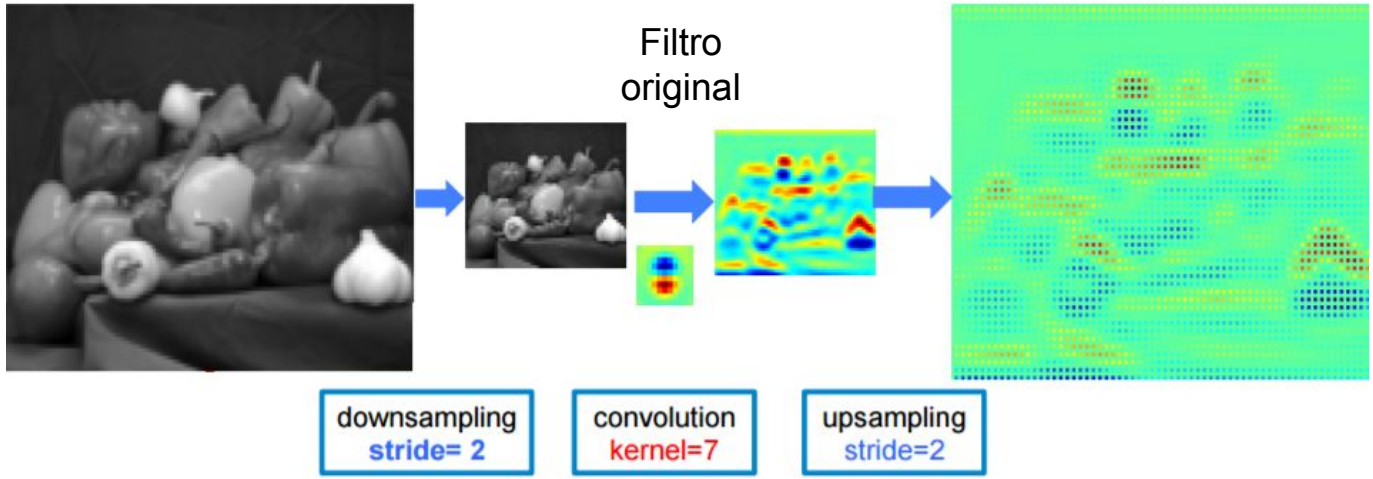
Convolução Dilatada - Padding



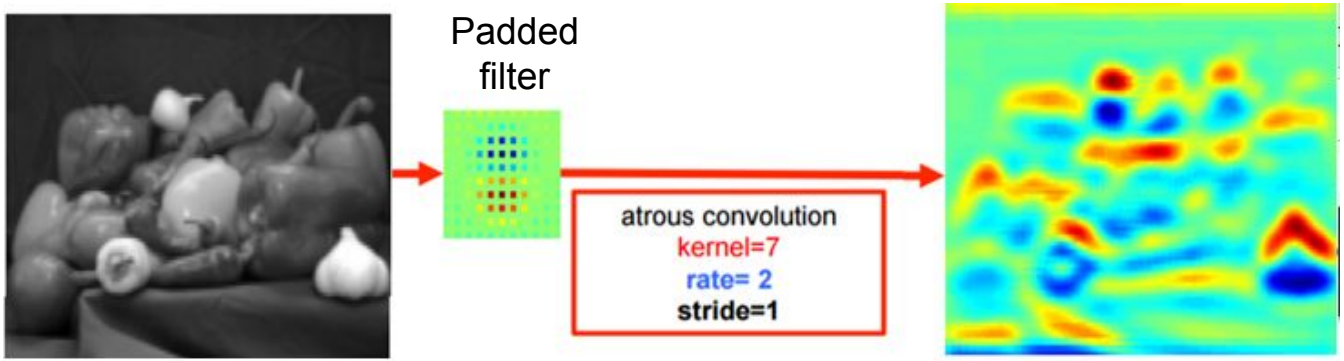
Dilated Convolution

Convolução Dilatada

Standard convolution

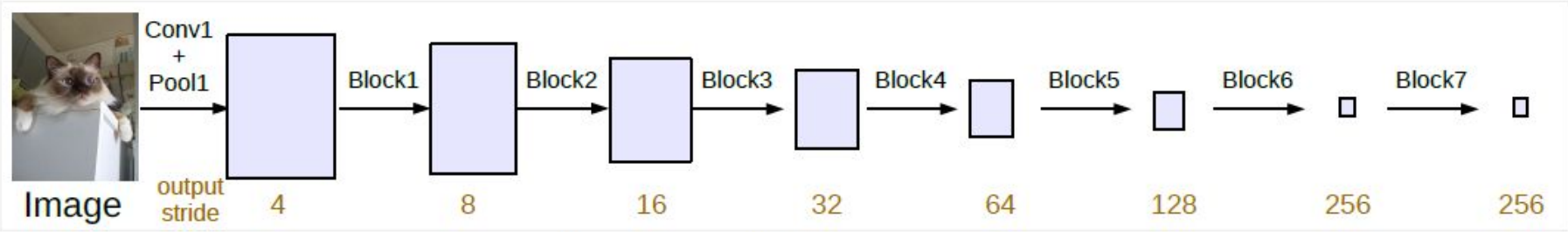


Atrous convolution

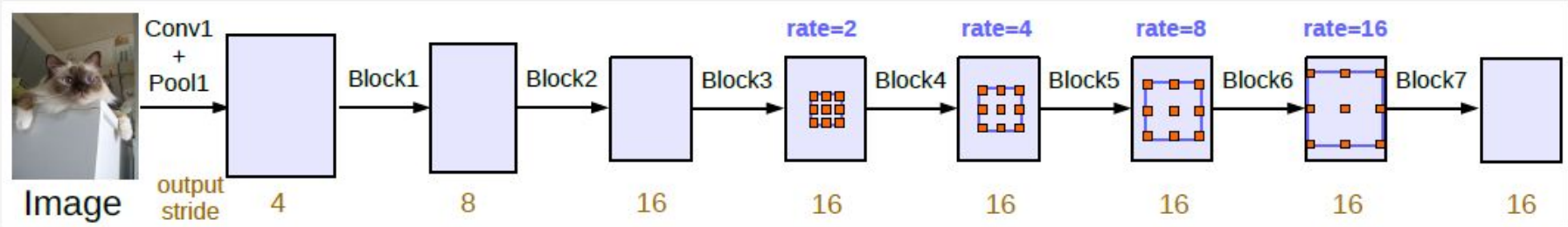


Convolução Dilatada

- Podemos manter o *stride* constante, mas com um *field-of-view* maior sem aumentar o número de parâmetros ou a quantidade de computação.
- Temos um *feature-map* maior, o que é bom para a segmentação semântica.

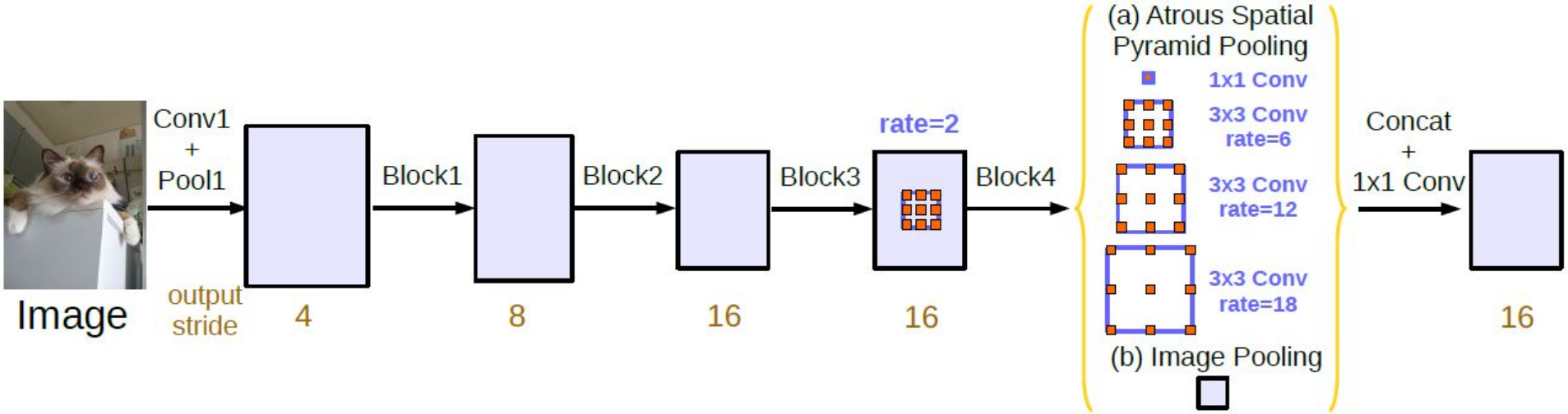


(a) Going deeper without atrous convolution.



(b) Going deeper with atrous convolution. Atrous convolution with $rate > 1$ is applied after block3 when $output\_stride = 16$.

Atrous Spatial Pyramid Pooling (ASPP)



Chen, L. C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2017). Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE transactions on pattern analysis and machine intelligence, 40(4), 834-848.