Detection and Segmentation

Stanford University

- https://www.youtube.com/watch?v=nDPWywWRIRo

Do sada: klasifikacija slika

So far: Image Classification

This image is CC0 public domain

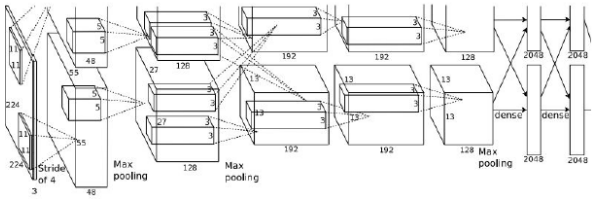


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

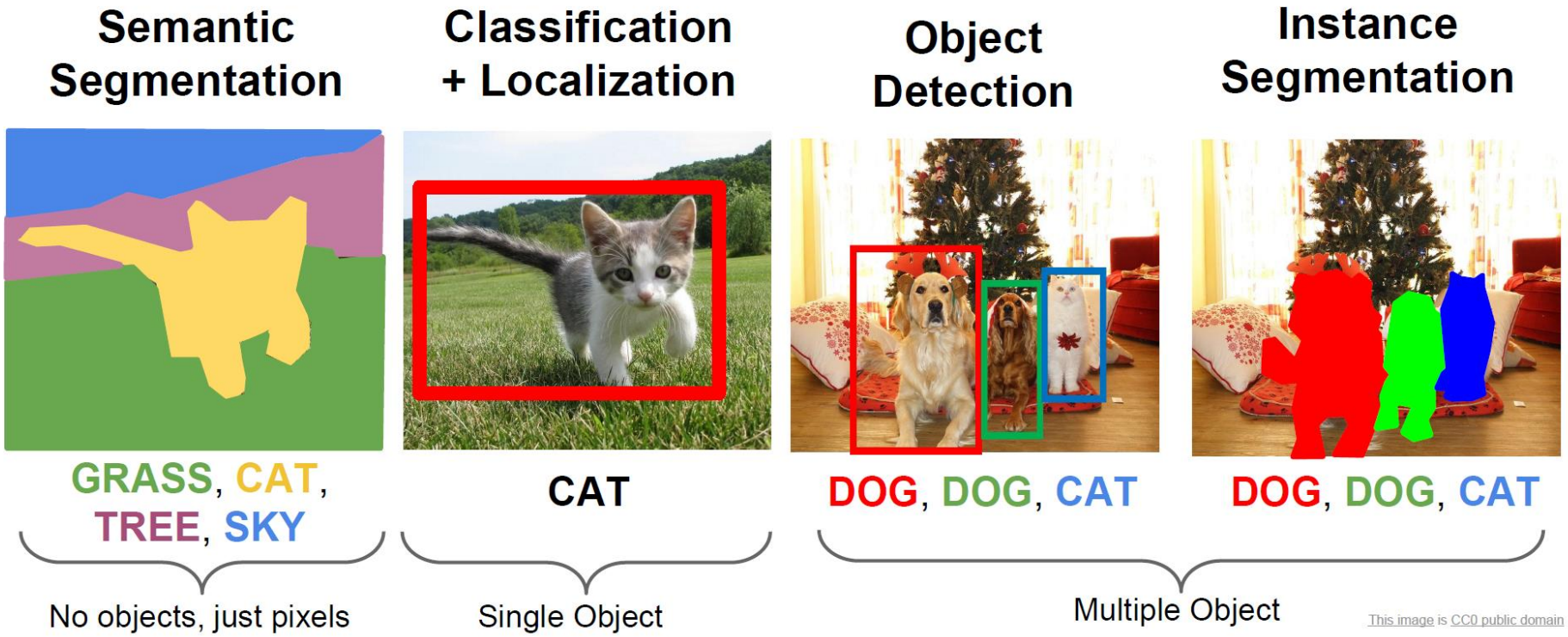
Vector:
4096

Fully-Connected:
4096 to 1000

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

Današnje predavanje

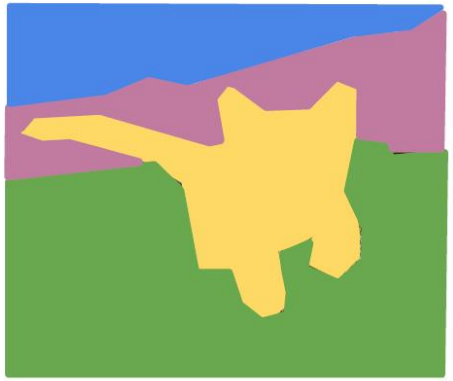
Other Computer Vision Tasks



Semantička segmentacija

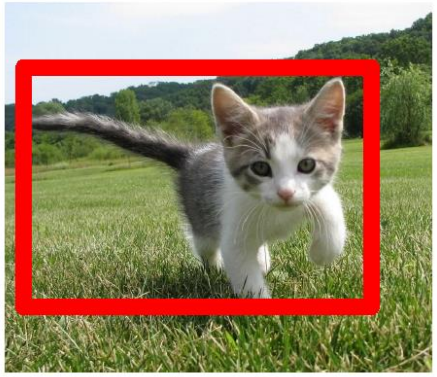
Semantic Segmentation

- Izlaz: svakom pikselu je dodeljena (jedna od predefinisanih) kategorija
- Skupi trening podaci



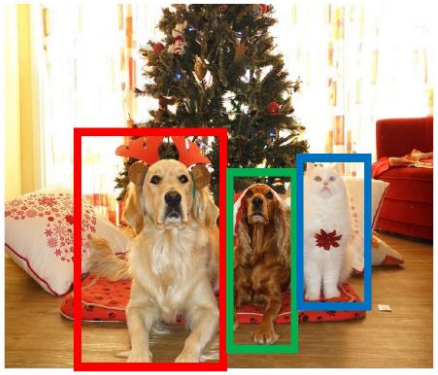
GRASS, CAT, TREE, SKY

No objects, just pixels

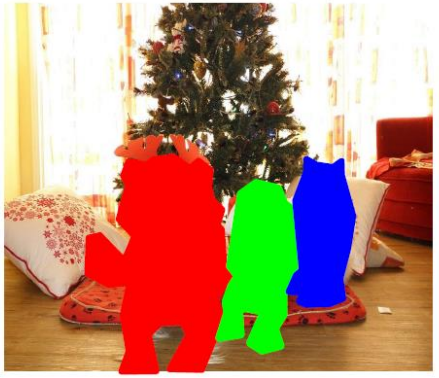


CAT

Single Object



DOG, DOG, CAT



DOG, DOG, CAT

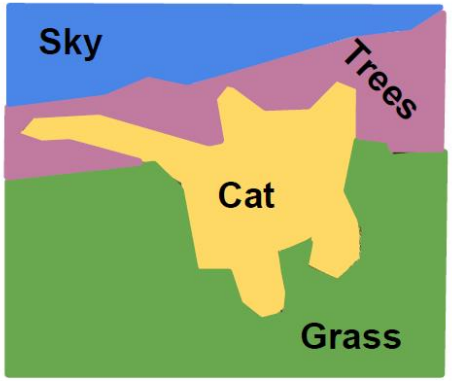
Multiple Object

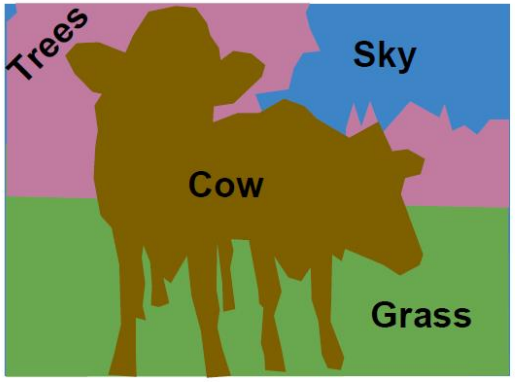
Semantička segmentacija

Semantic Segmentation

Label each pixel in the image with a category label

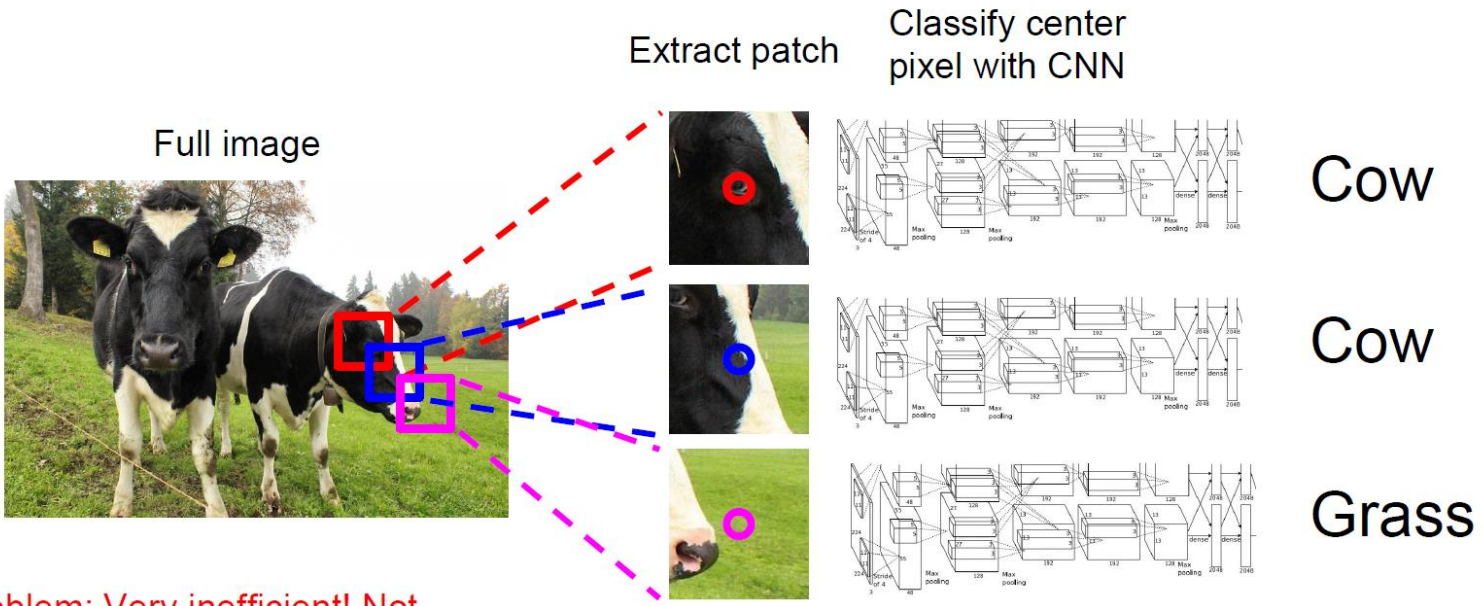
Don’t differentiate instances, only care about pixels





Pristup: primeniti klasifikaciju

Semantic Segmentation Idea: Sliding Window



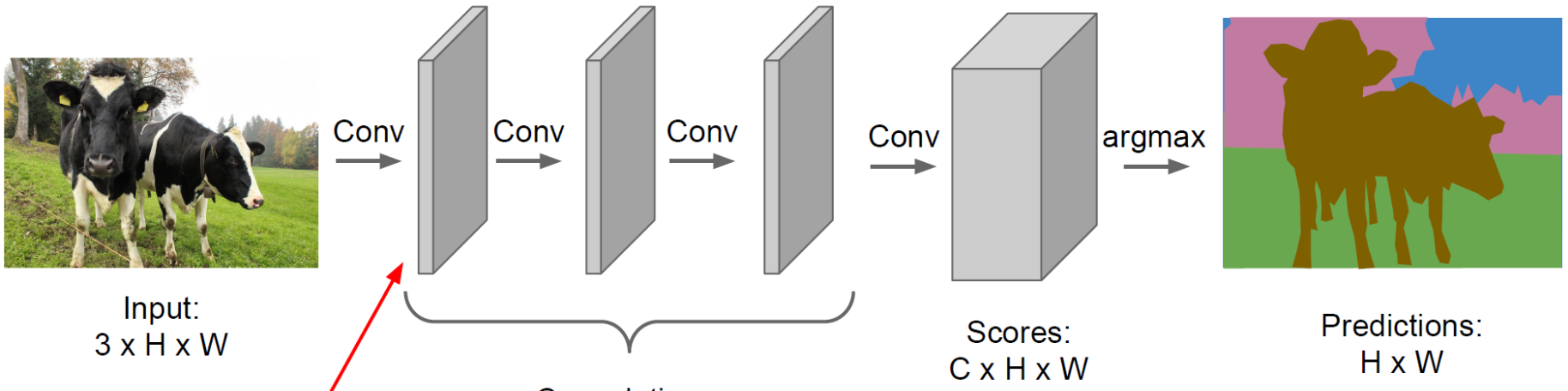
Problem: Very inefficient! Not reusing shared features between overlapping patches

Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013
Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014

Izbacivanje potpuno povezanih slojeva

Semantic Segmentation Idea: Fully Convolutional

Design a network as a bunch of convolutional layers to make predictions for pixels all at once!

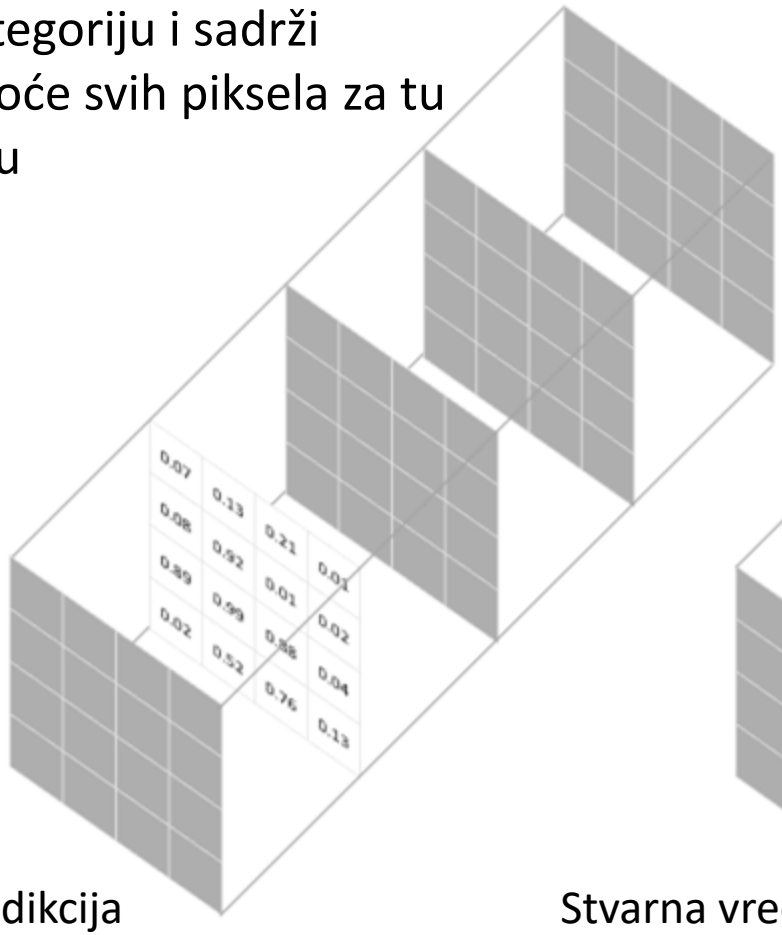


Problem: convolutions at original image resolution will be very expensive ...

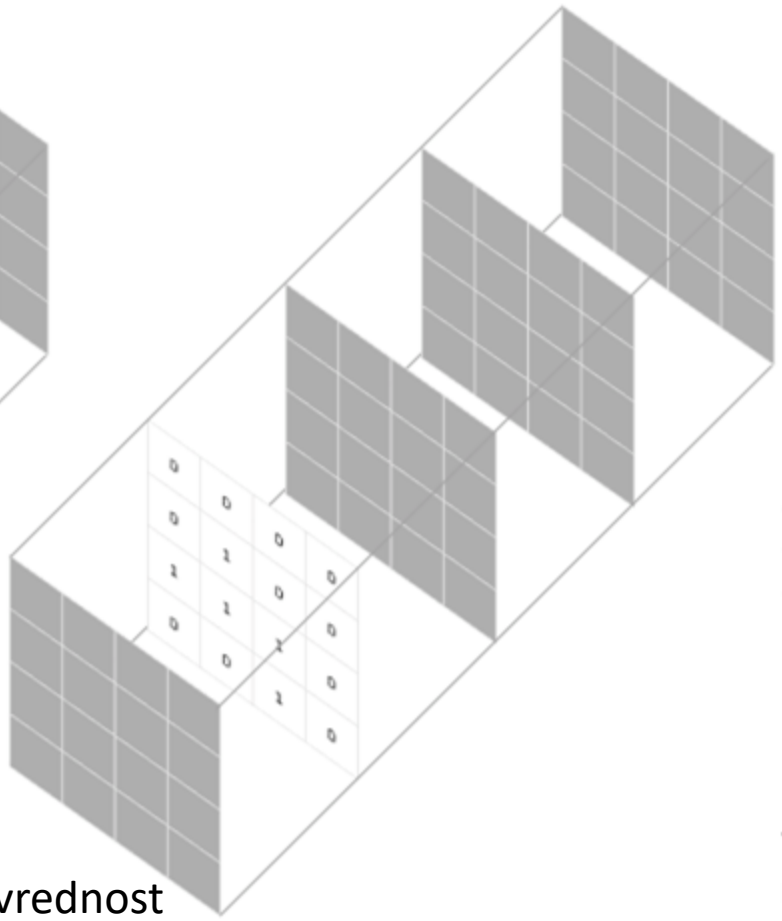
- Očuvavamo prostornu rezoluciju
- C – broj kategorija u našem sistemu

Izlaz iz *softmax* na nivou kategorije

- C=5: dubina izlaznog sloja je 5
- Jedan *feature map* predstavlja jednu kategoriju i sadrži verovatnoće svih piksela za tu kategoriju



Predikcija

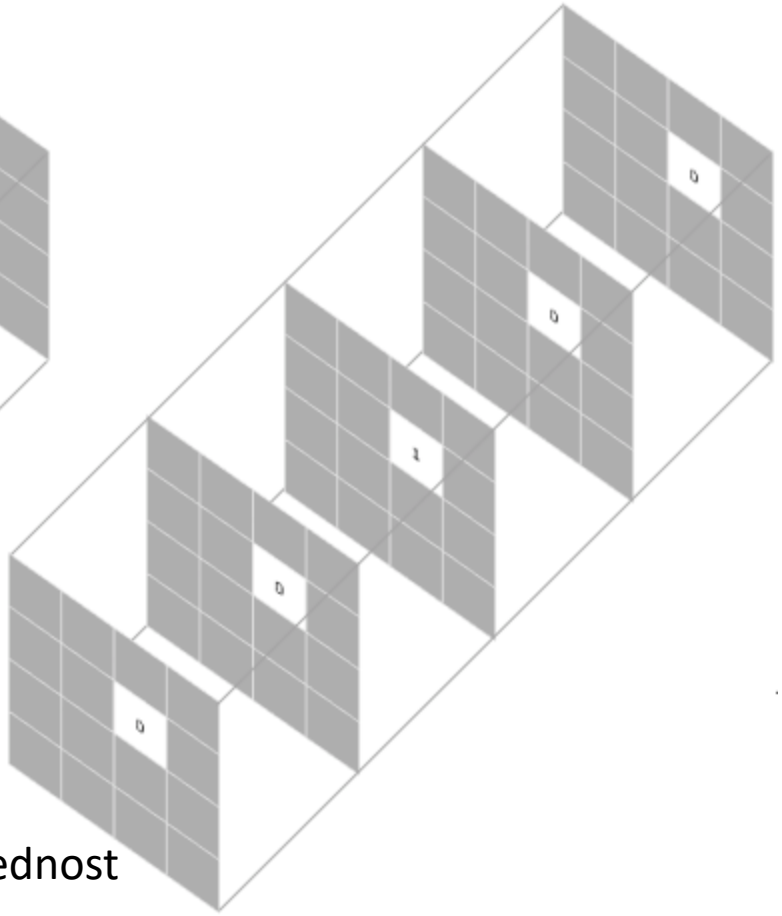


Stvarna vrednost

Izlaz iz *softmax* za jedan piksel

- *Softmax* će za svaku od 5 kategorija da izračuna verovatnoću da piksel pripada toj kategoriji (ukupan zbir verovatnoća za sve kategorije je 1)
- U primeru: 0.02 + 0.01 + 0.83 + 0.08 + 0.06 = 1.
- Pošto 3. kategorija ima najveću verovatnoću (0.83), za piksel bi bila prediktovana 3. kategorija

Predikcija



Stvarna vrednost

Računanje gubitka (*loss*)

- U prethodnom primeru za dati piksel:
  - Predikcija:

$$y_{pred} = [0.02, 0.01, 0.83, 0.08, 0.06]$$

  - Stvarna vrednost (piksel pripada kategoriji 3):

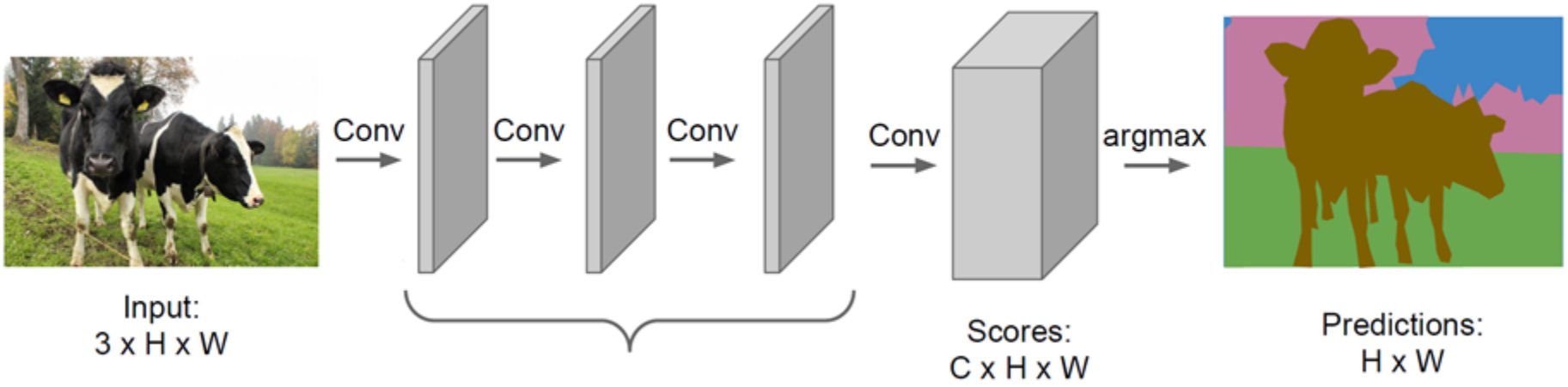
$$y_{true} = [0, 0, 1, 0, 0]$$

  - Računa se *cross-entropy loss*:

$$-\sum_{classes} y_{true} \log y_{pred}$$

- Za pojedinačnu sliku u trening skupu se uproseči gubitak za sve piksele na njoj
- Ukupan gubitak za trening skup se aproksimira uprosečavanjem gubitka za *mini batch*
- Treniramo propagacijom unazad

Izbacivanje potpuno povezanih slojeva



- Relativno jednostavan model koji bi verovatno radio razumno dobro

- Problem:
  - Primenjujemo puno konvolucija, od kojih sve očuvavaju prostorne dimenzije, što čini postupak veoma skupim
    - Ako npr. želite konvolucije koje imaju 64, 128 ili 256 filtera (dosta često u CNN), a rezolucija ulaza je velika, ovo zauzima veliku količinu memorije
  - Zato u praksi ne vidimo često ovaj tip arhitekture

U praksi: *downsampling* pa *upsampling*

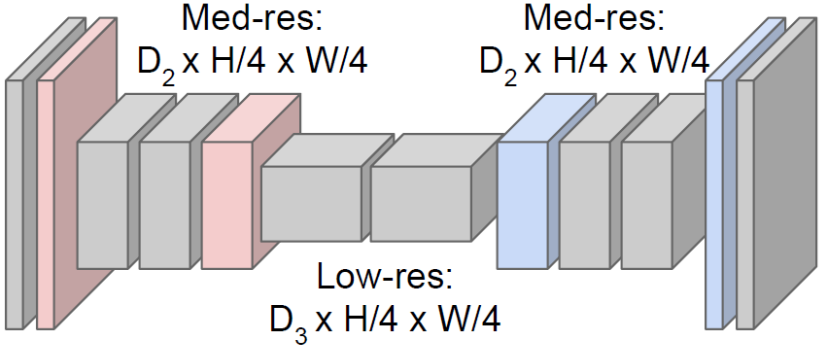
Semantic Segmentation Idea: Fully Convolutional

Downsampling:
Pooling, strided convolution

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

Upsampling:
???

Input:
3 x H x W



High-res:
D_1 x H/2 x W/2

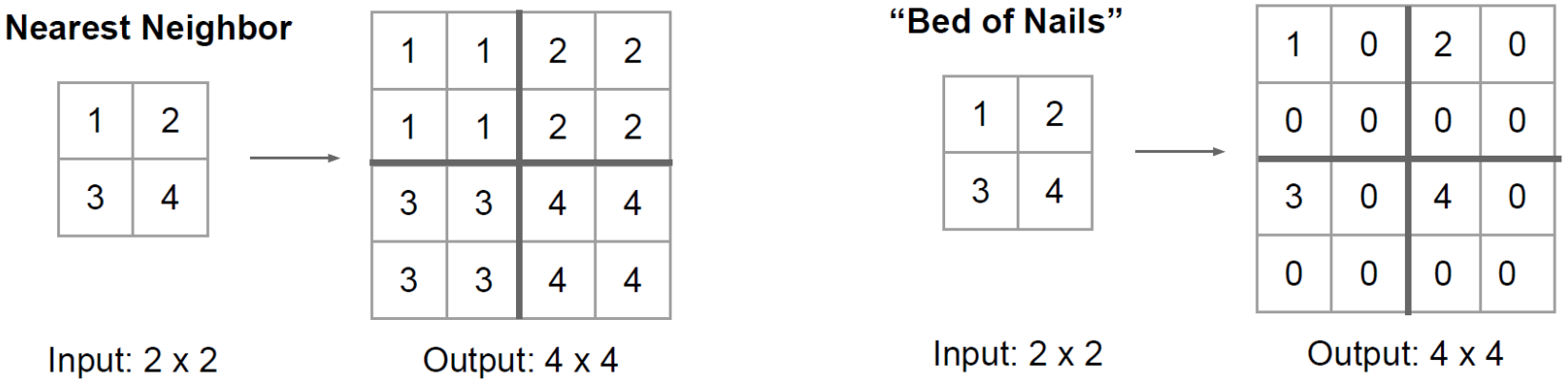
High-res:
D_1 x H/2 x W/2

Predictions:
H x W

Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015

Jedna strategija za *upsampling*: *unpooling*

In-Network upsampling: “Unpooling”



Simetrija

In-Network upsampling: “Max Unpooling”

Max Pooling
Remember which element was max!

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

Input: 4 x 4

5	6
7	8

Output: 2 x 2

. . .

Rest of the network

Max Unpooling
Use positions from pooling layer

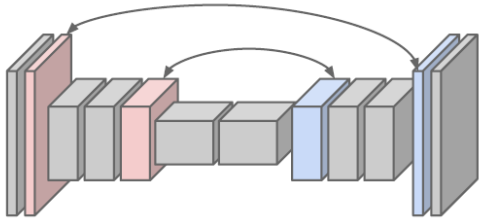
1	2
3	4

Input: 2 x 2

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

Output: 4 x 4

Corresponding pairs of downsampling and upsampling layers

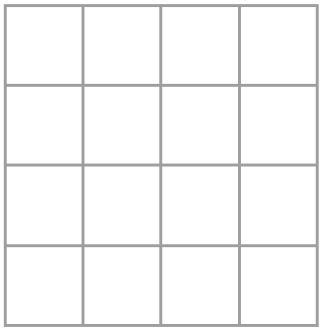


Druga strategija: *Transpose Convolution*

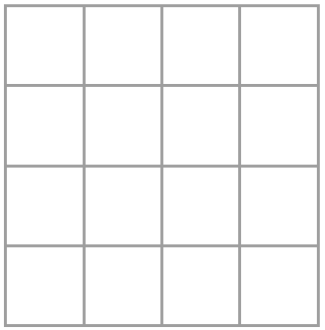
- *Nearest Neighbor* i *Bed of Nails* su fiksirane funkcije – ne „uče“ kako da rade upsampling

Learnable Upsampling: Transpose Convolution

Recall: Typical 3 x 3 convolution, stride 1 pad 1



Input: 4 x 4

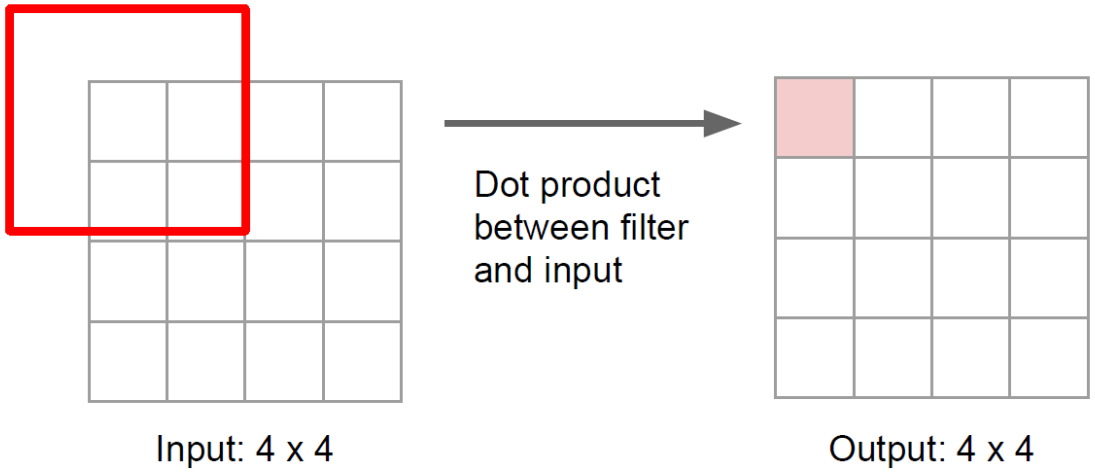


Output: 4 x 4

Druga strategija: *Transpose Convolution*

Learnable Upsampling: Transpose Convolution

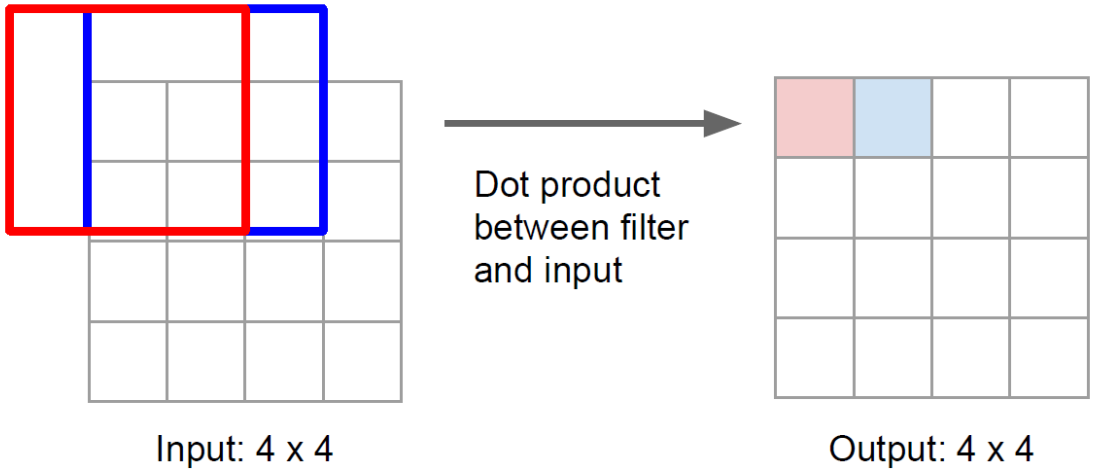
Recall: Normal 3 x 3 convolution, stride 1 pad 1



Druga strategija: *Transpose Convolution*

Learnable Upsampling: Transpose Convolution

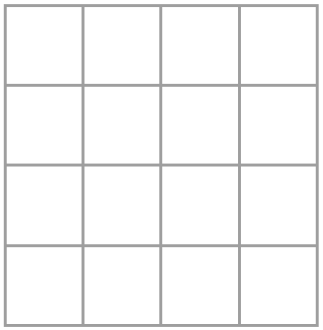
Recall: Normal 3 x 3 convolution, stride 1 pad 1



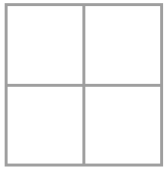
Druga strategija: *Transpose Convolution*

Learnable Upsampling: Transpose Convolution

Recall: Normal 3 x 3 convolution, stride 2 pad 1



Input: 4 x 4

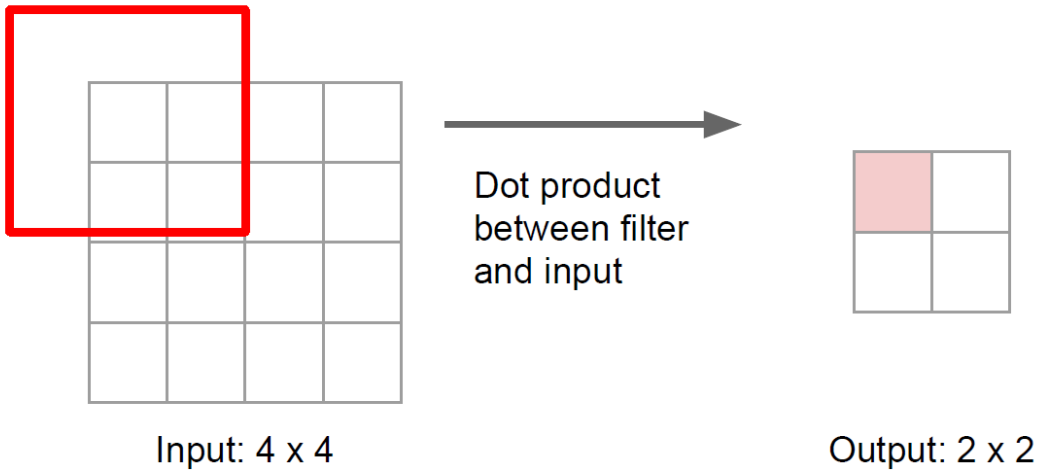


Output: 2 x 2

Druga strategija: *Transpose Convolution*

Learnable Upsampling: Transpose Convolution

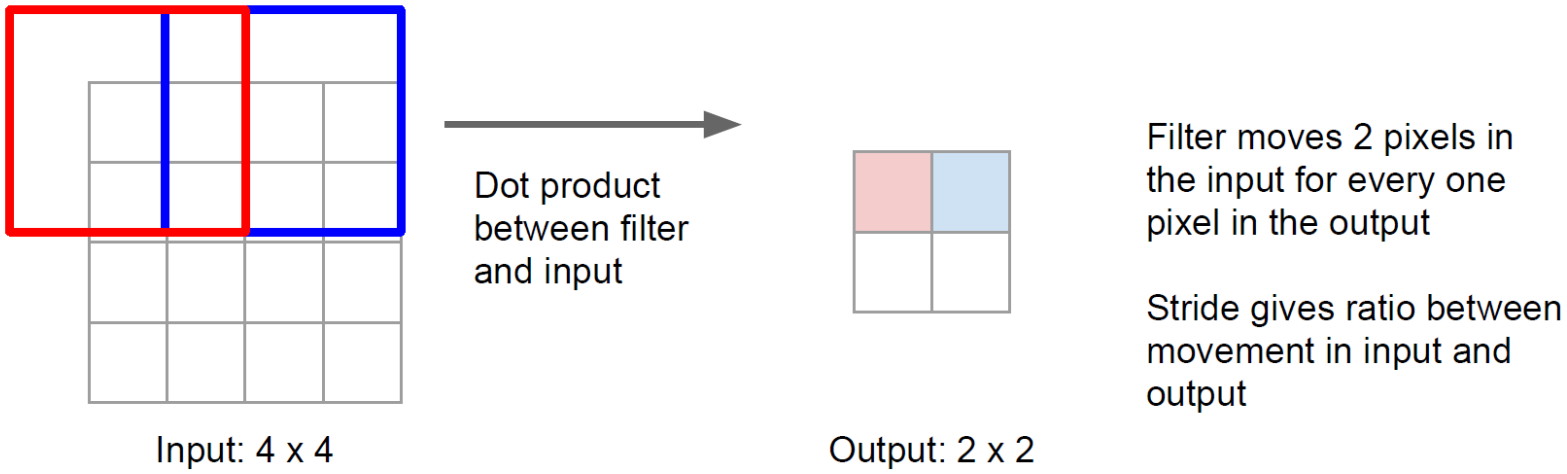
Recall: Normal 3 x 3 convolution, stride 2 pad 1



Druga strategija: *Transpose Convolution*

Learnable Upsampling: Transpose Convolution

Recall: Normal 3 x 3 convolution, stride 2 pad 1

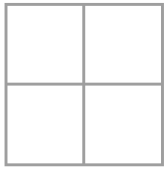


- U ovom slučaju: *downsample*-uje se *feature map* 2 puta, na način koji se može naučiti (učimo težine)

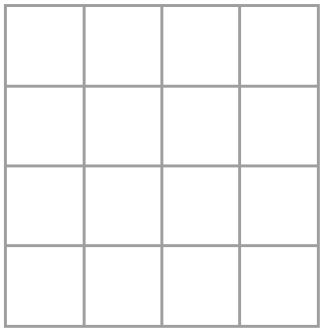
Druga strategija: *Transpose Convolution*

Learnable Upsampling: Transpose Convolution

3 x 3 **transpose** convolution, stride 2 pad 1



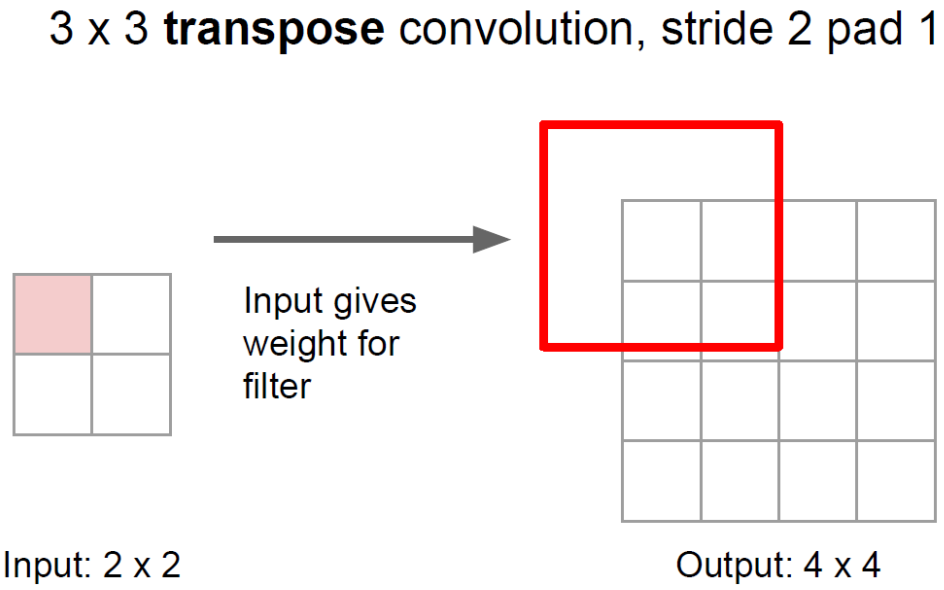
Input: 2 x 2



Output: 4 x 4

Druga strategija: *Transpose Convolution*

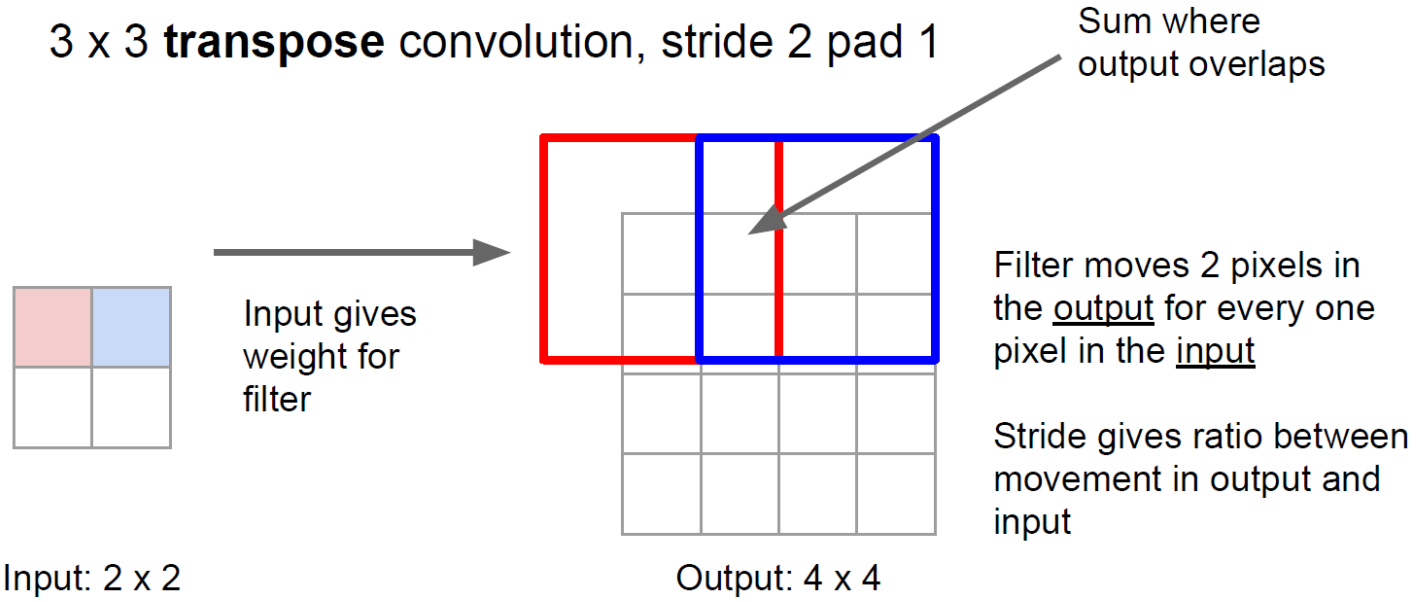
Learnable Upsampling: Transpose Convolution



- Treniranjem ćemo naučiti težine u filteru
- Ulaz u upsampling predstavlja težinu kojom se naučeni filter množi
- Otežinjene vrednosti fitera se kopiraju na odgovarajuće mesto u izlazu

Druga strategija: *Transpose Convolution*

Learnable Upsampling: Transpose Convolution

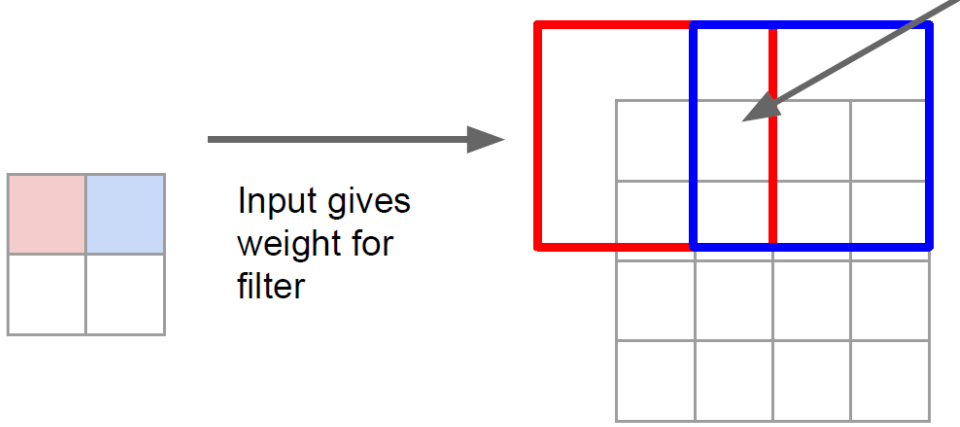


Druga strategija: *Transpose Convolution*

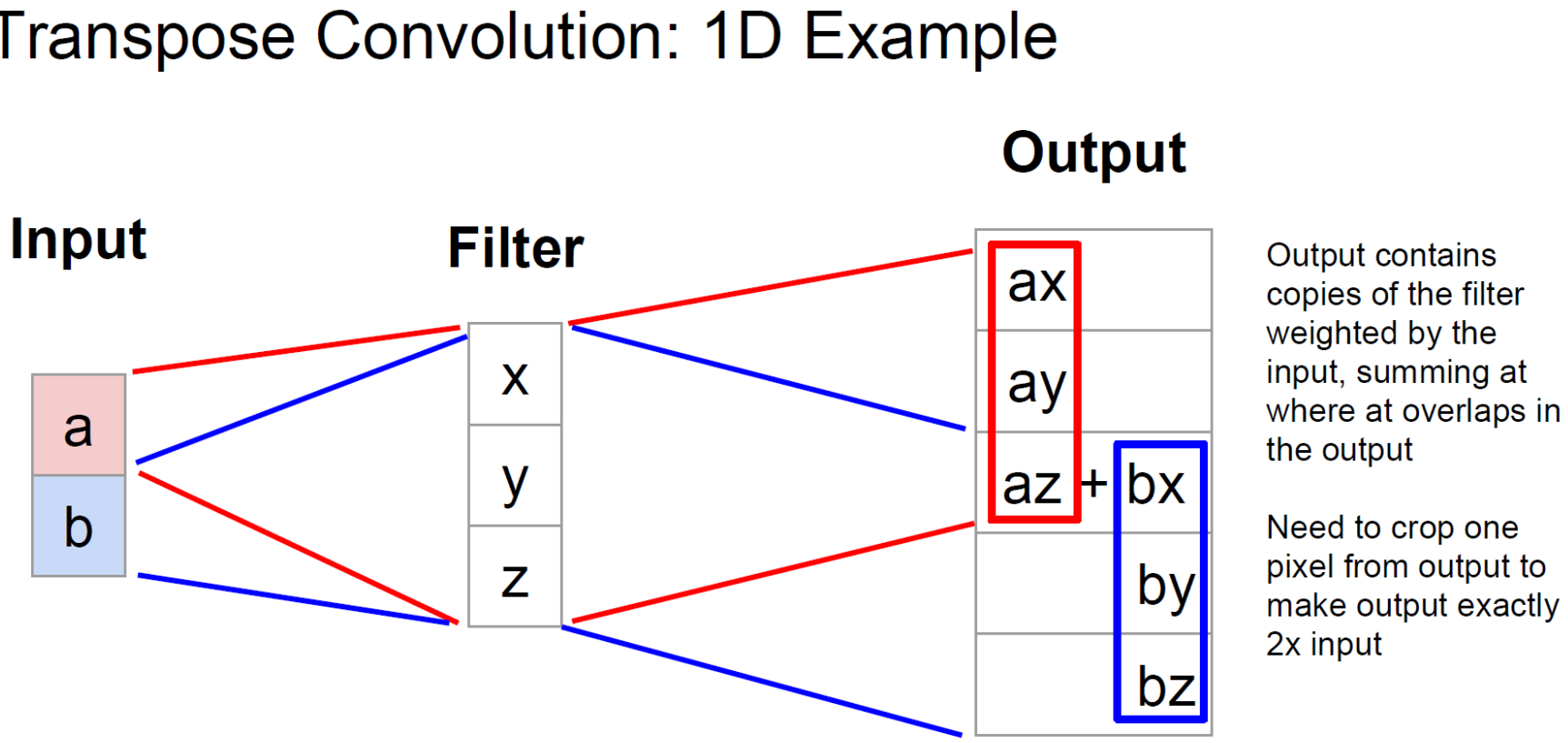
Learnable Upsampling: Transpose Convolution

Other names:
-Deconvolution (bad)
-Upconvolution
-Fractionally strided convolution
-Backward strided convolution

3 x 3 **transpose** convolution, stride 2 pad 1



Druga strategija: *Transpose Convolution*



Zašto „transpose convolution“?

Convolution as Matrix Multiplication (1D Example)

We can express convolution in terms of a matrix multiplication

$$\vec{x} * \vec{a} = X\vec{a} \quad x = [x, y, z] \quad a = [a, b, c, d]$$

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

Example: 1D conv, kernel size=3, stride=1, padding=1

Zašto „transpose convolution“?

Convolution as Matrix Multiplication (1D Example)

We can express convolution in terms of a matrix multiplication

$$\vec{x} * \vec{a} = X\vec{a} \quad x = [x, y, z] \quad a = [a, b, c, d]$$

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

Example: 1D conv, kernel size=3, stride=1, padding=1

Convolution transpose multiplies by the transpose of the same matrix:

$$\vec{x} *^T \vec{a} = X^T \vec{a}$$

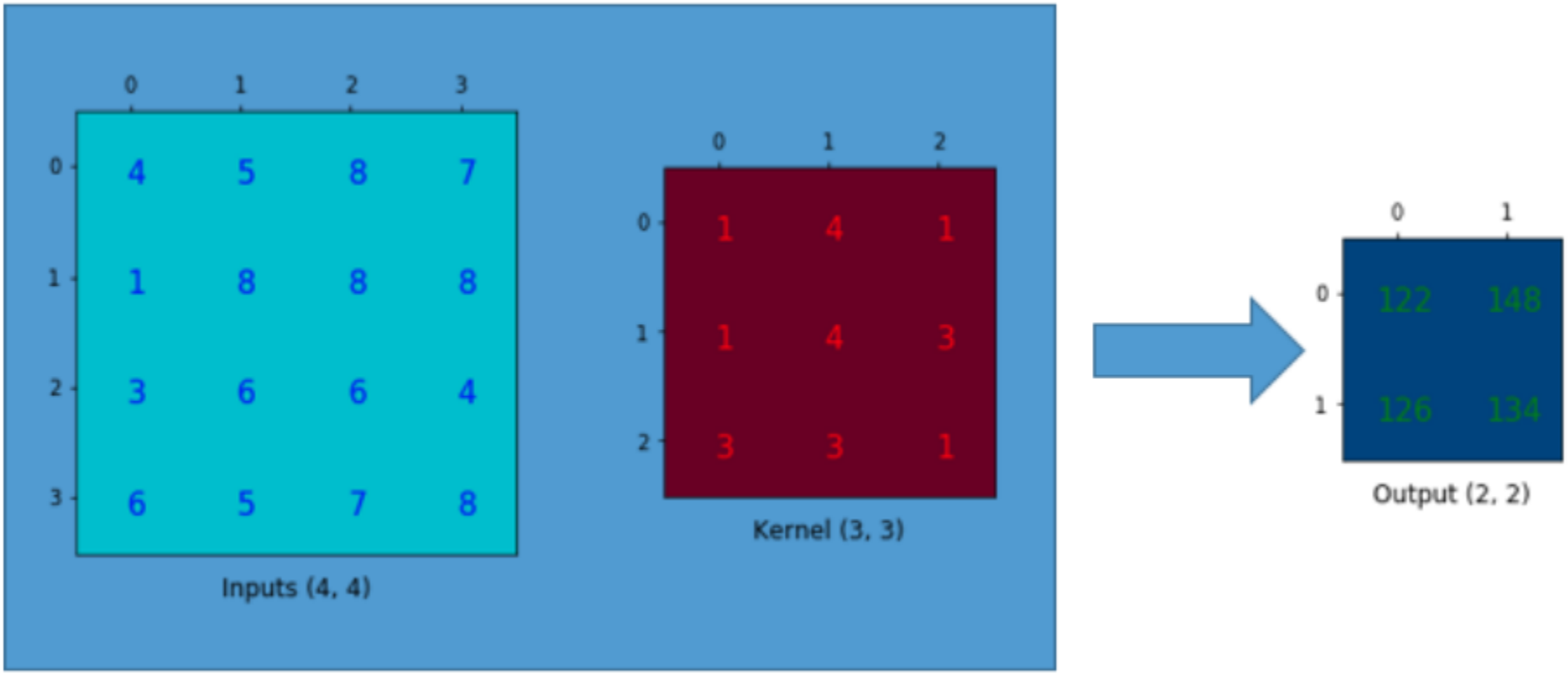
$$\begin{bmatrix} x & 0 & 0 & 0 \\ y & x & 0 & 0 \\ z & y & x & 0 \\ 0 & z & y & x \\ 0 & 0 & z & y \\ 0 & 0 & 0 & z \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} ax \\ ay + bx \\ az + by + cx \\ bz + cy + dx \\ cz + dy \\ dz \end{bmatrix}$$

When stride=1, convolution transpose is just a regular convolution (with different padding rules)

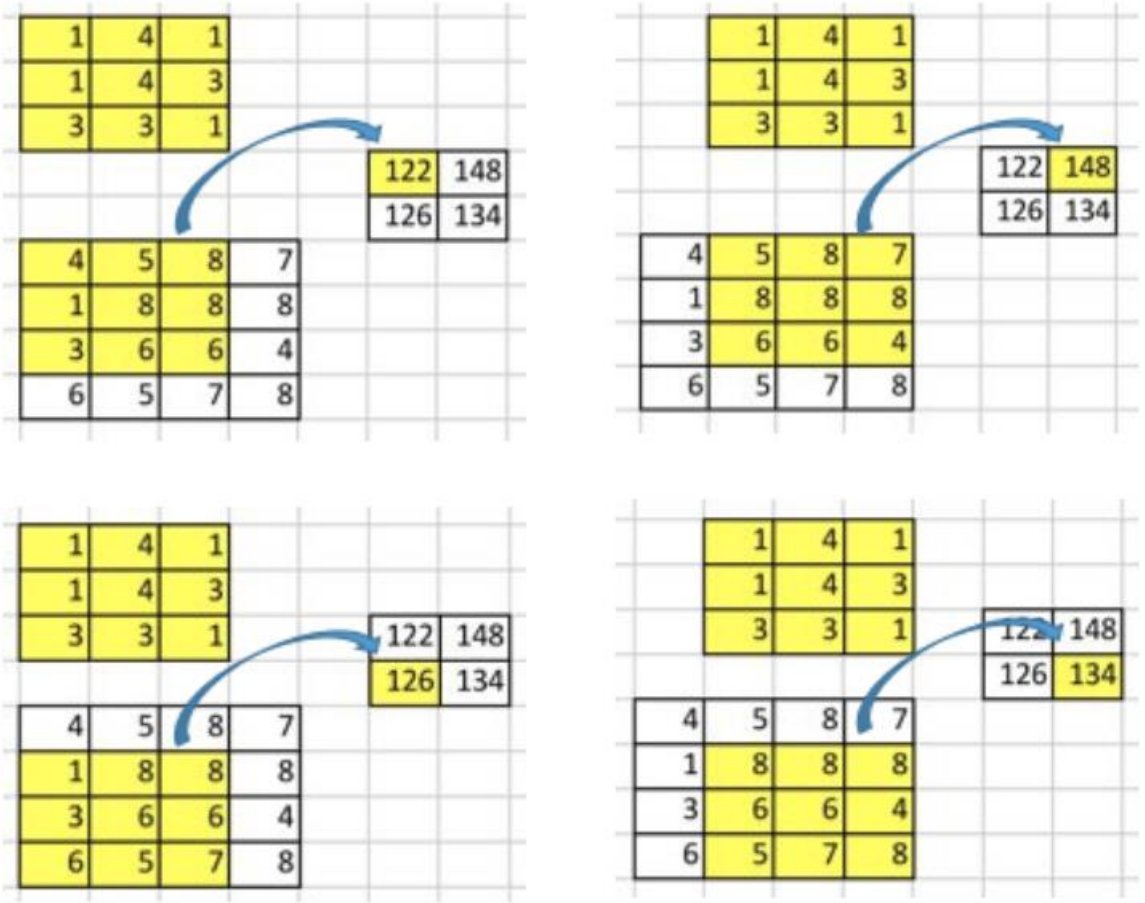
Primer 2D konvolucija

- Ulaz 4×4, filter 3×3
- nema *zero padding*-a
- korak (*stride*) je 1

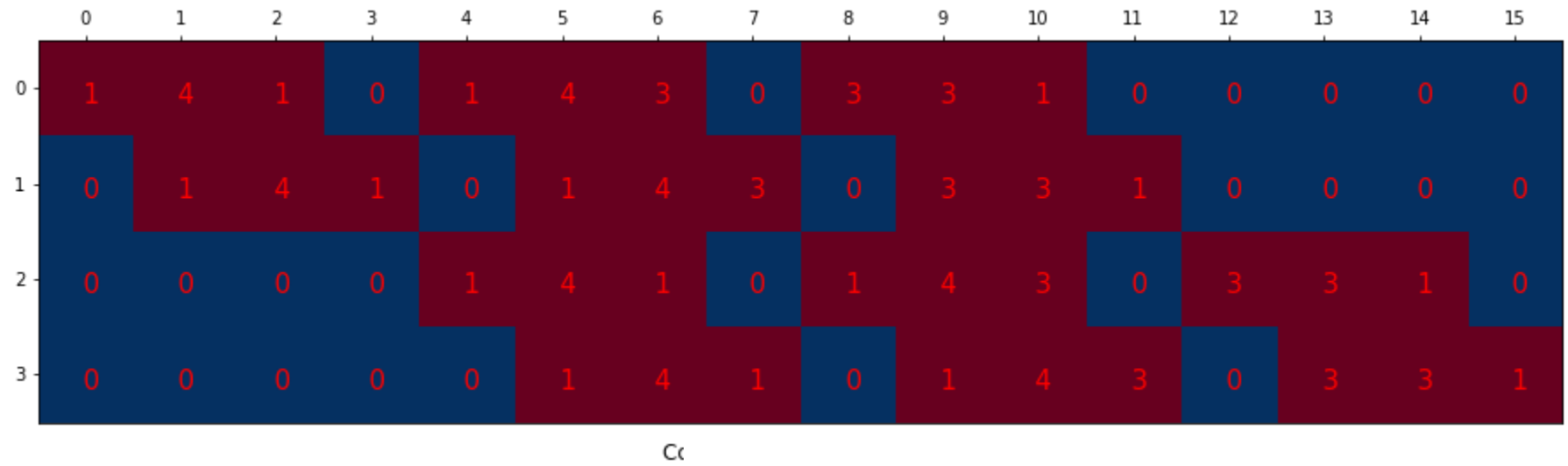
https://towardsdatascience.com/up-sampling-with-transposed-convolution-9ae4f2df52d0



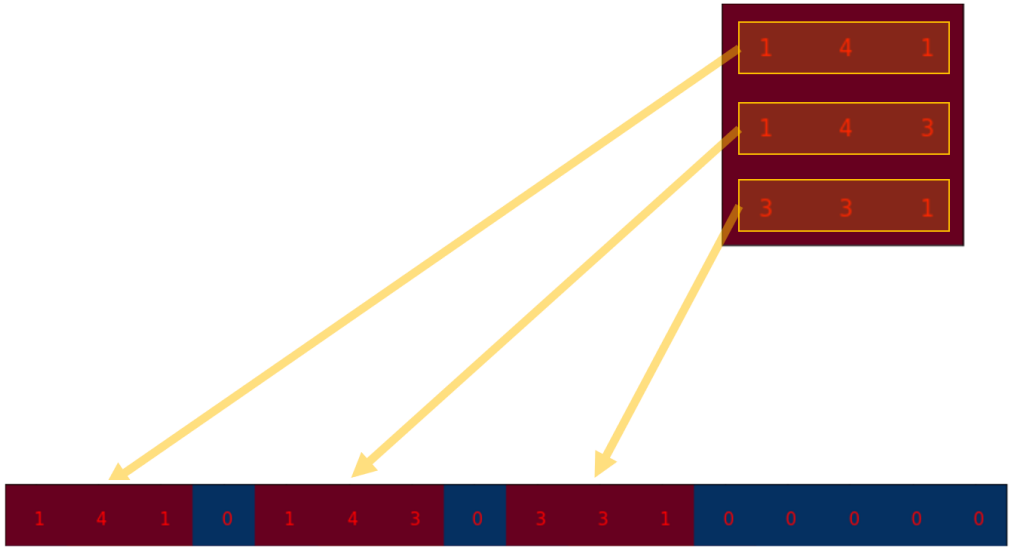
Primer 2D konvolucija



Konvolucija izražena kao množenje matrica

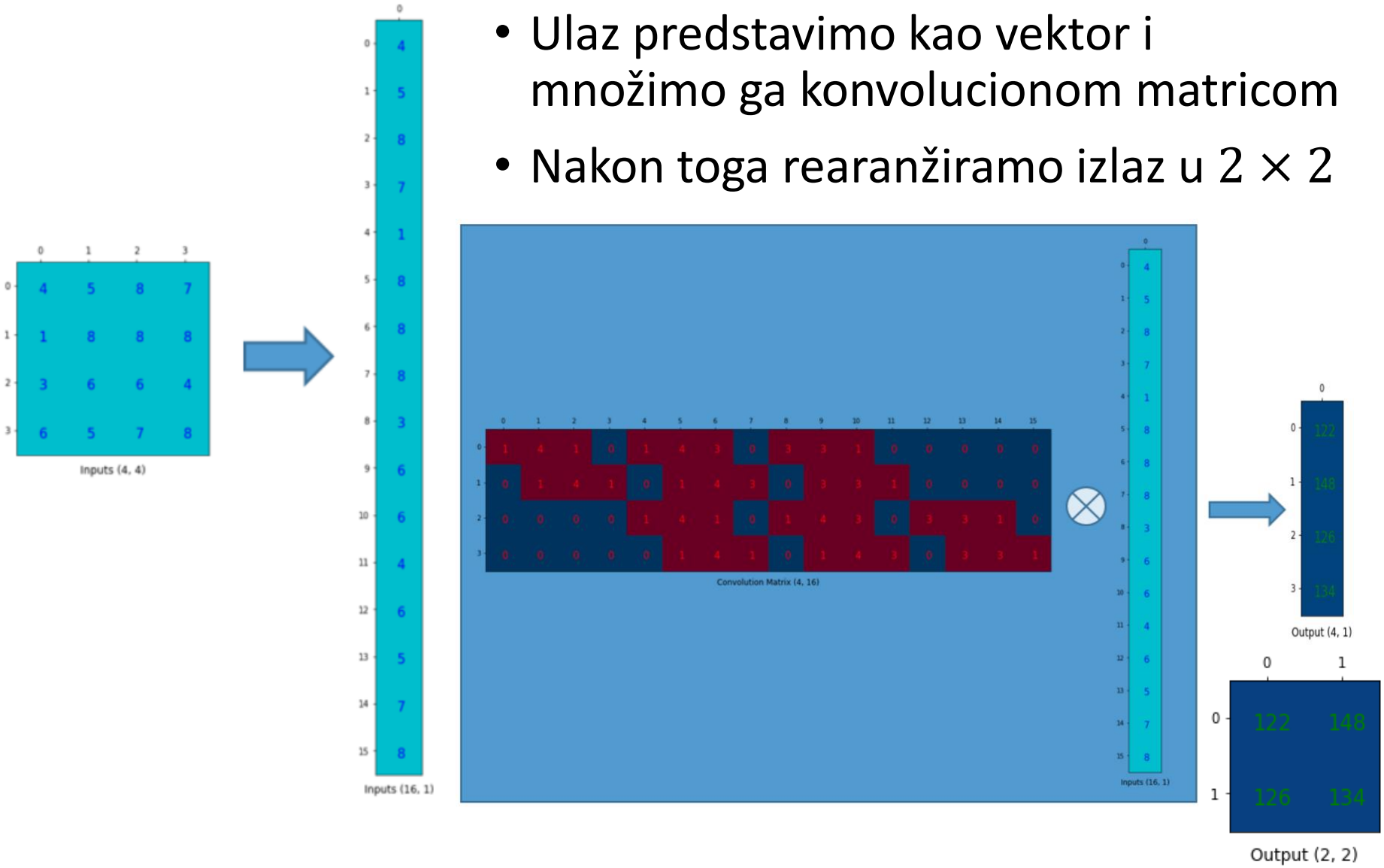


- Svaki red konvolucione matrice je aranžiran filter sa *zero padding*-om

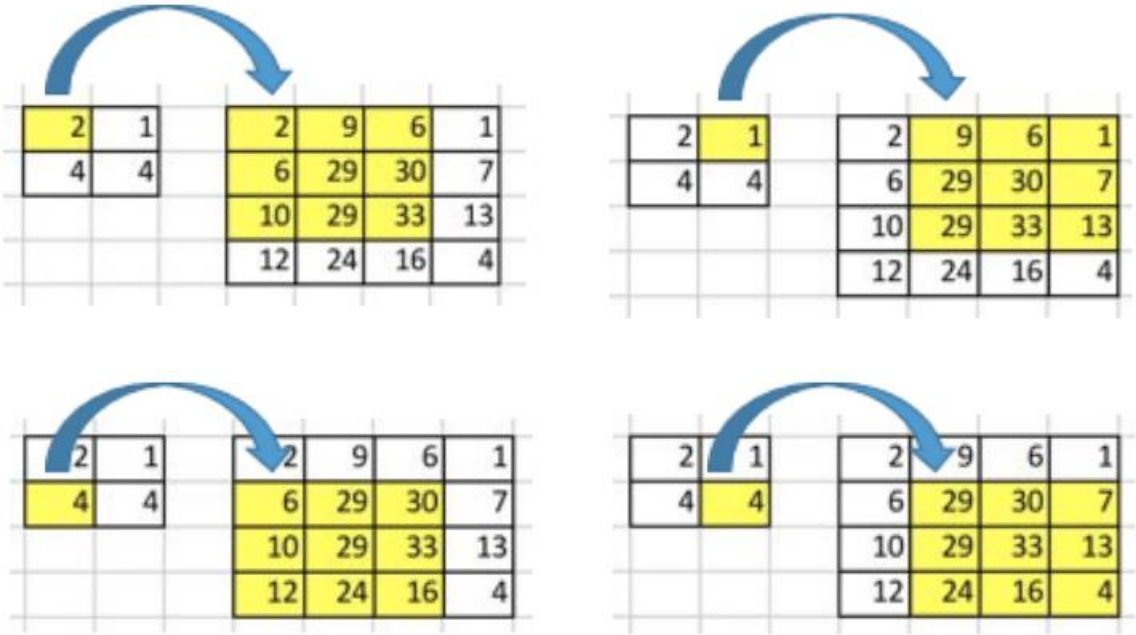


Konvolucija izražena kao množenje matrica

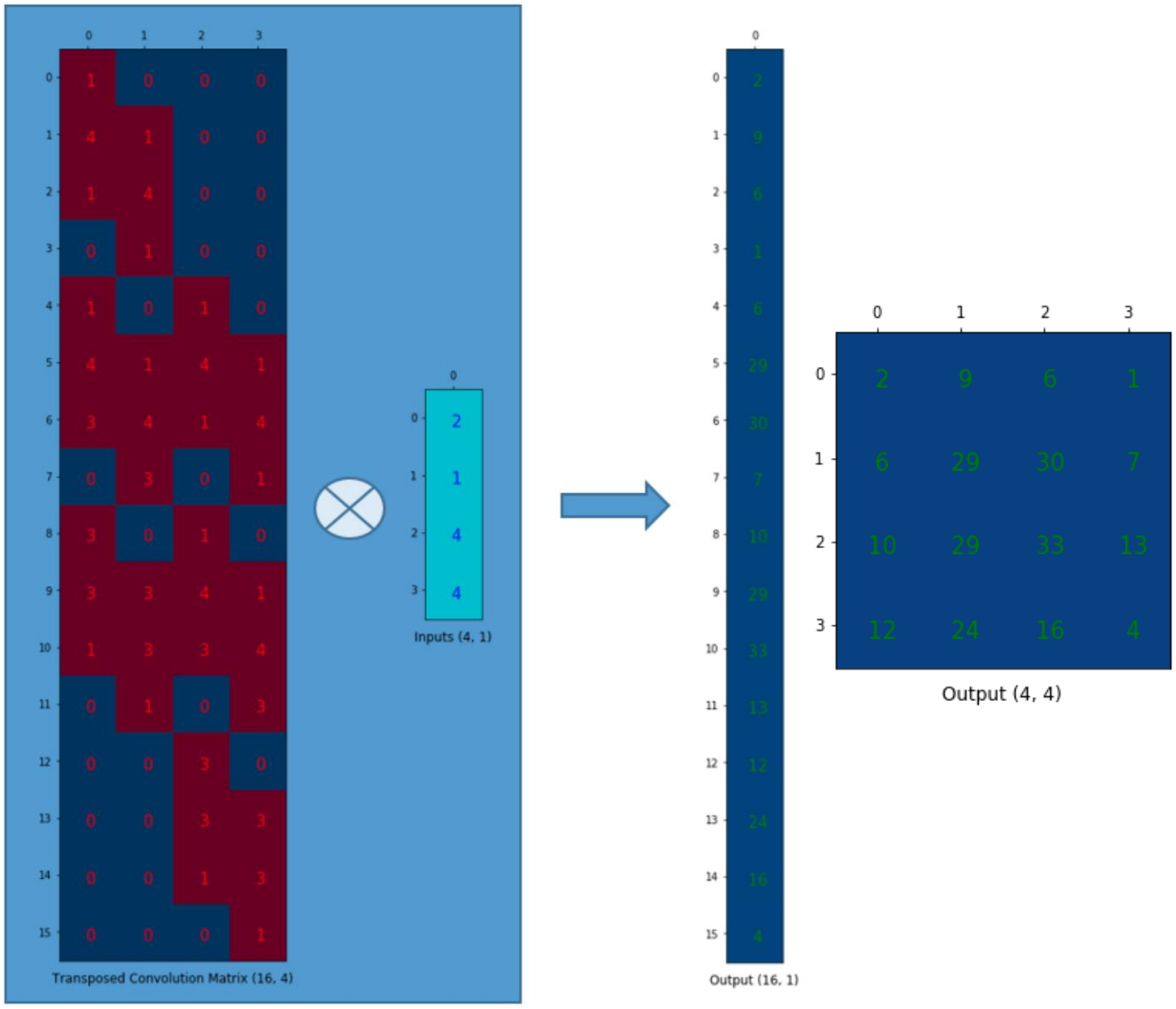
- Ulaz predstavimo kao vektor i množimo ga konvolucionom matricom
- Nakon toga rearanžiramo izlaz u 2×2



Transponova konvolucija



Transponovana konvolucija



- Ulaz 2×2 smo mapirali na 4×4
- Stvarne vrednosti u matrici ne moraju biti one iz originalne konvolucione matrice
- Ovo je samo ilustracija da se upsampling može izraziti kao množenje matrica u obrnutom maniru od konvolucije
- Težine u transponovanoj konvoluciju ćemo naučiti (nemamo predefinisani metod interpolacije nedostajućih vrednosti kao u *Nearest Neighbor*)

Stride = 2

Convolution as Matrix Multiplication (1D Example)

We can express convolution in terms of a matrix multiplication

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

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

Example: 1D conv, kernel size=3, stride=2, padding=1

Stride = 2

Convolution as Matrix Multiplication (1D Example)

We can express convolution in terms of a matrix multiplication

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

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

Example: 1D conv, kernel size=3, stride=2, padding=1

Convolution transpose multiplies by the transpose of the same matrix:

$$\vec{x} *^T \vec{a} = X^T \vec{a}$$

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

When stride>1, convolution transpose is no longer a normal convolution!

Potpuna konvoluciona mreža

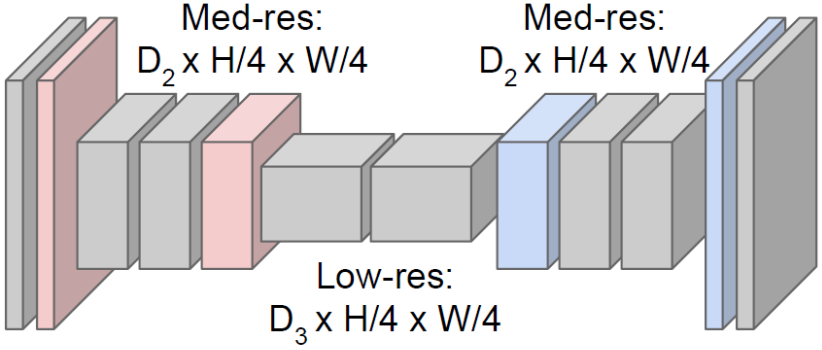
Semantic Segmentation Idea: Fully Convolutional

Downsampling:
Pooling, strided convolution

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

Upsampling:
Unpooling or strided transpose convolution

Input:
3 x H x W



High-res:
D1 x H/2 x W/2

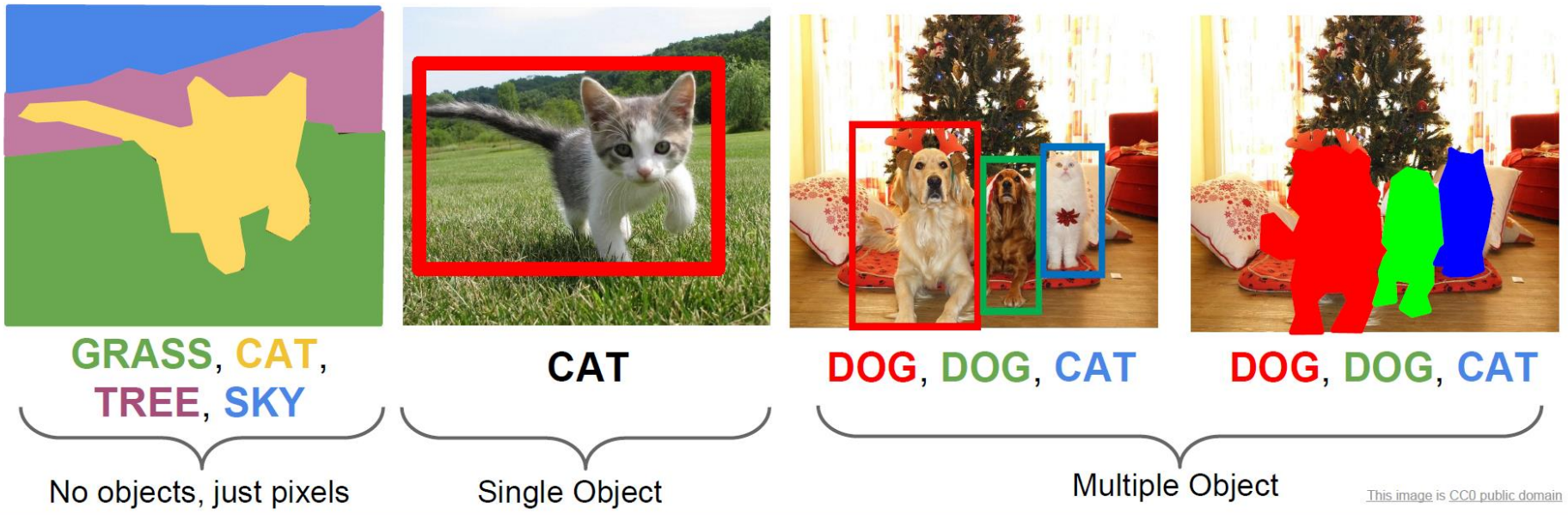
High-res:
D1 x H/2 x W/2

Predictions:
H x W

Long, Shelhamer, and Darrell, "Fully Convolutional Networks for Semantic Segmentation", CVPR 2015
Noh et al, "Learning Deconvolution Network for Semantic Segmentation", ICCV 2015

Lokalizacija

- Pored predikcije kategorije, treba da odredimo gde se objekat nalazi na slici (vratimo tačno jedan *bounding box*)
- Razlika u odnosu na detekciju objekta: unapred pretpostavljamo da ima jedna instanca objekta na slici koju ćemo klasifikovati
- Znamo da ćemo imati klasifikacionu odluku na slici i vraćamo jedan *bounding box*

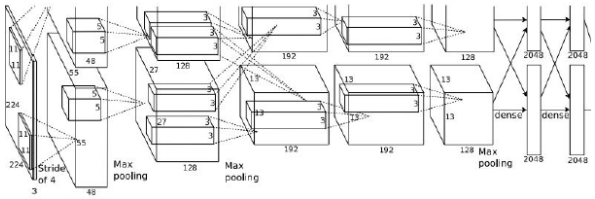


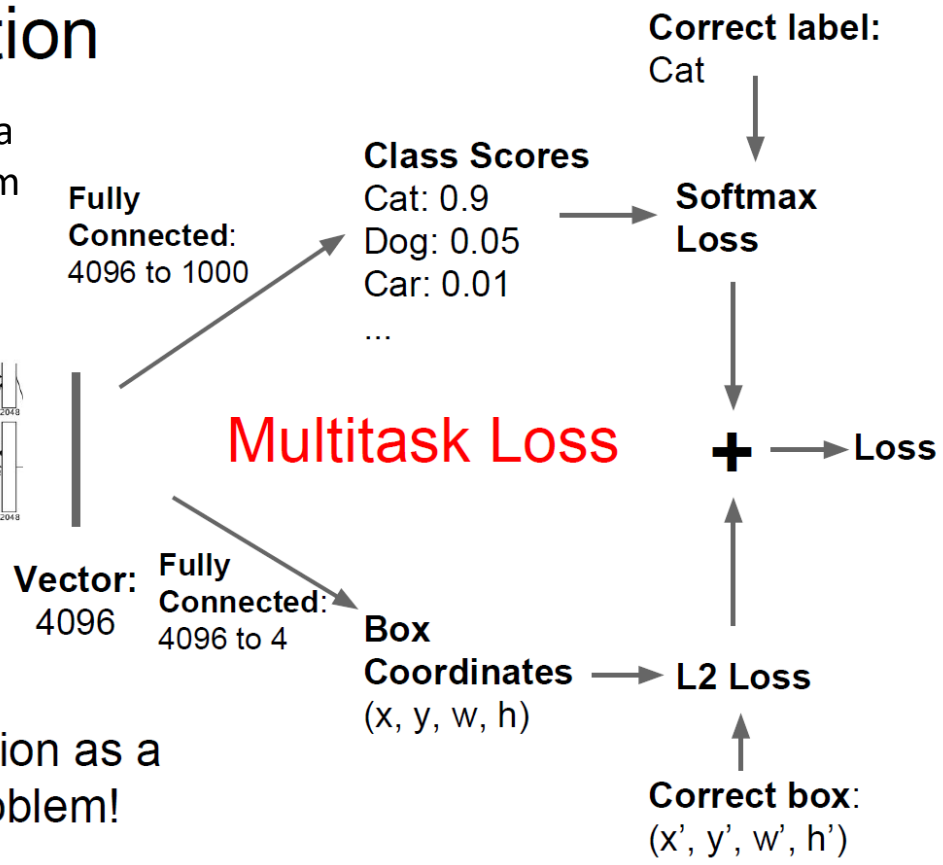
Osnovna arhitektura

Classification + Localization

- Pretpostavljamo *fully supervised setting*: svaka slika iz trening skupa je anotirana i kategorijom i koordinatama *bounding box*-a

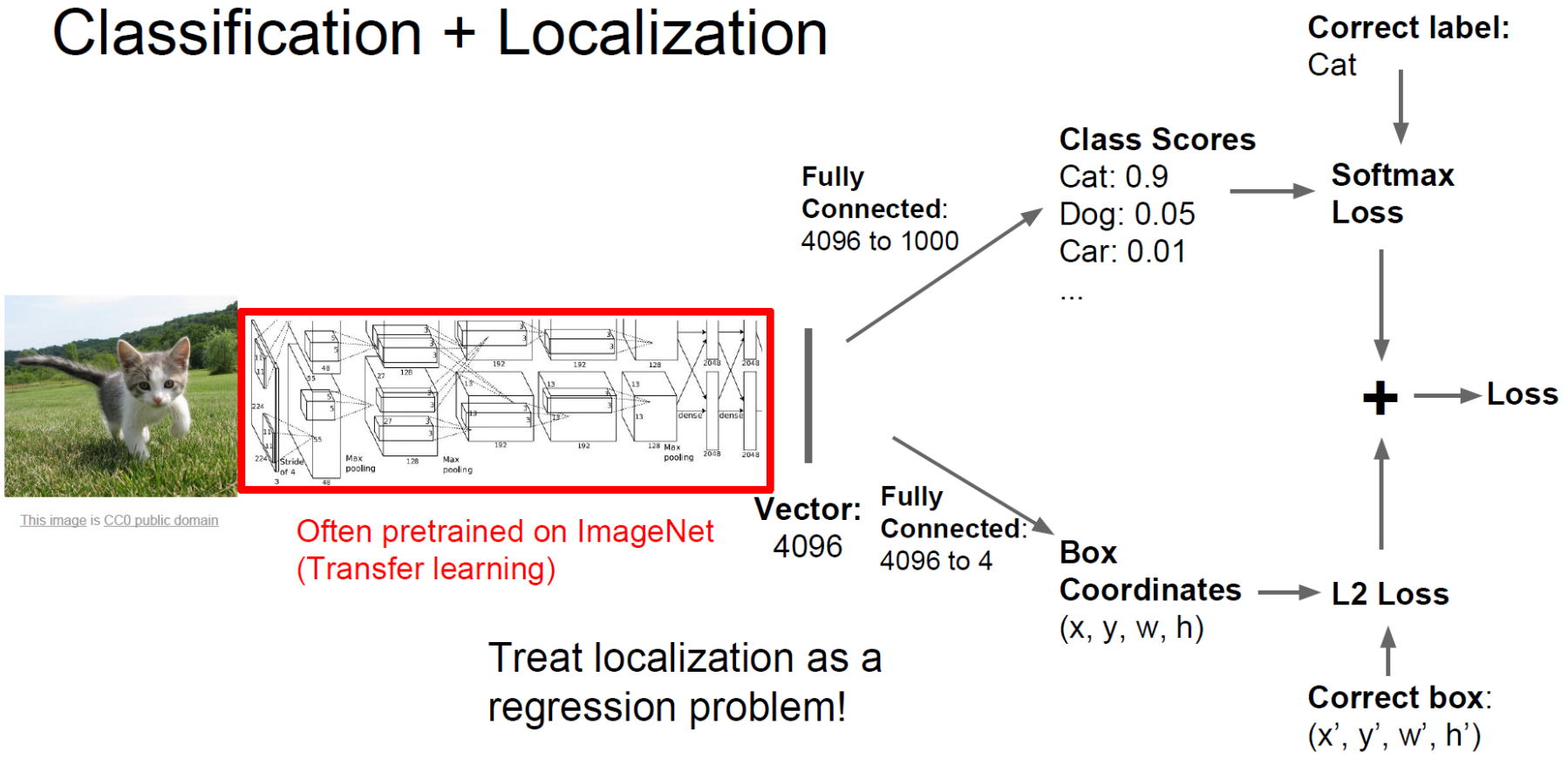
This image is CC0 public domain





Treat localization as a regression problem!

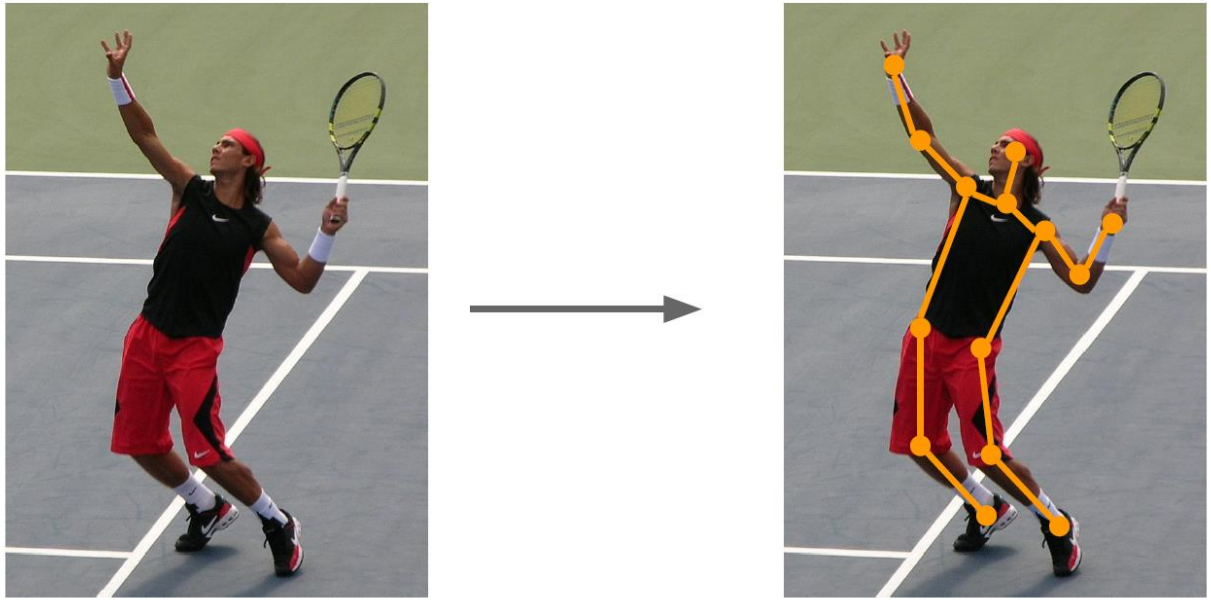
Osnovna arhitektura



Human Pose Estimation

- Ova ideja za predviđanje nekog fiksiranog broja pozicija na slici može da se koristi i u drugim problemima, ne samo za klasifikaciju + lokalizaciju

Aside: Human Pose Estimation



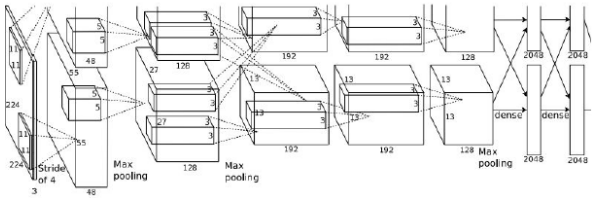
Represent pose as a set of 14 joint positions:

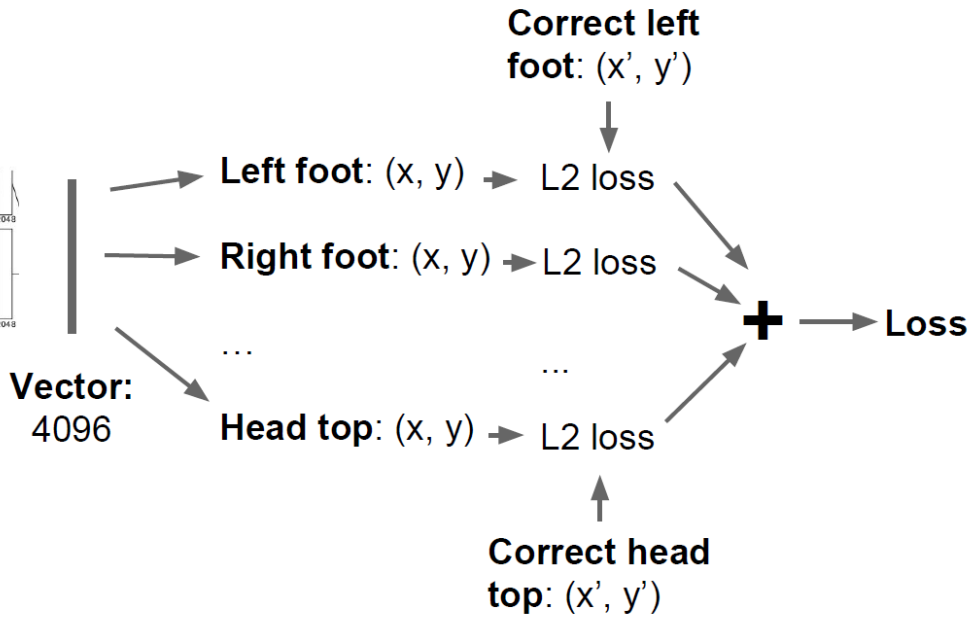
Left / right foot
Left / right knee
Left / right hip
Left / right shoulder
Left / right elbow
Left / right hand
Neck
Head top

Johnson and Everingham, "Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation", BMVC 2010

Human Pose Estimation

Aside: Human Pose Estimation

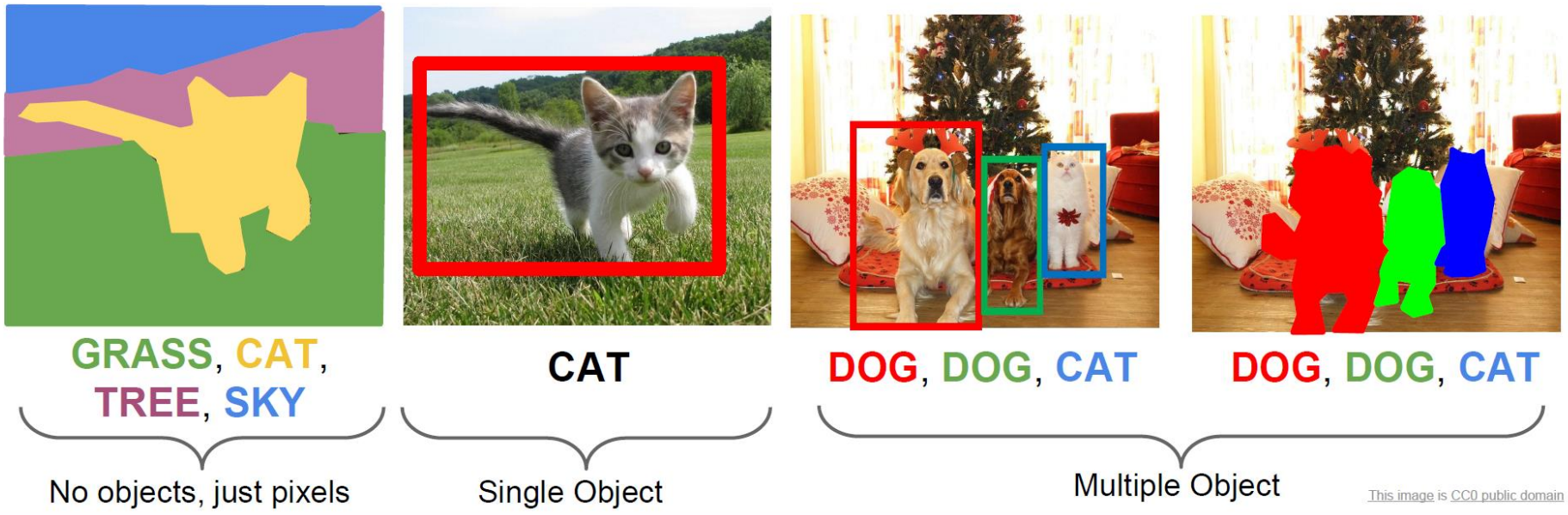




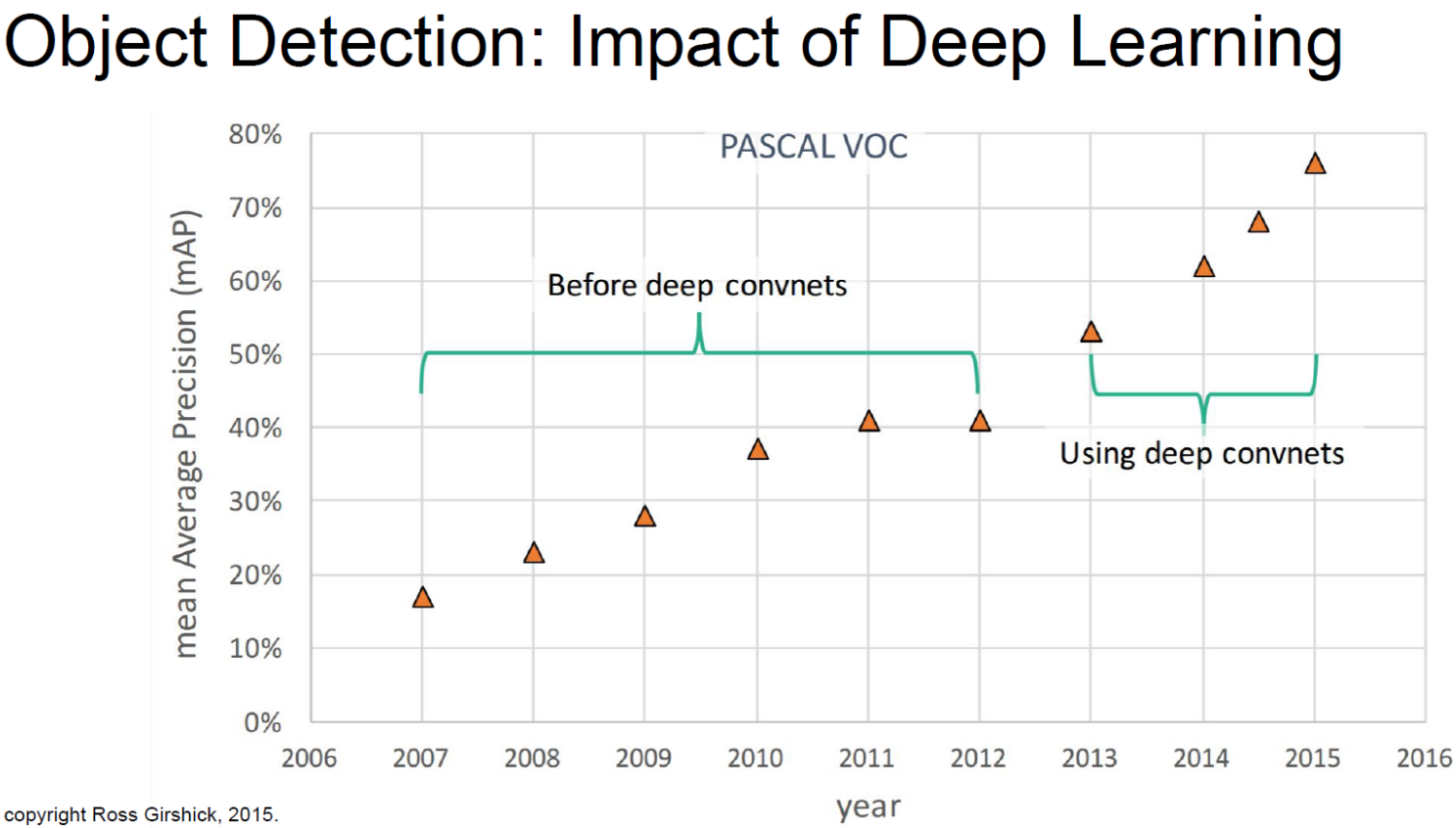
Toshev and Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks", CVPR 2014

Detekcija objekata

- Razlikuje se od *classification + localization* jer može da ima različit broj izlaza za svaku ulaznu sliku (ne znamo unapred koliko objekata očekujemo na slici)



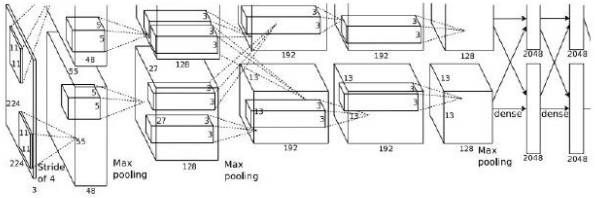
Detekcija objekata na PASCAL VOC



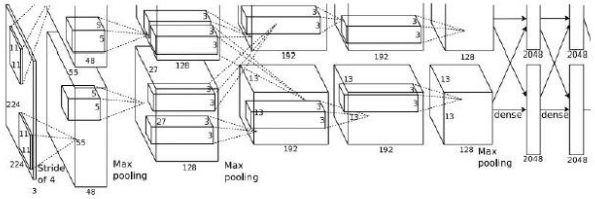
Možemo li formulisati kao problem regresije?

Object Detection as Regression?

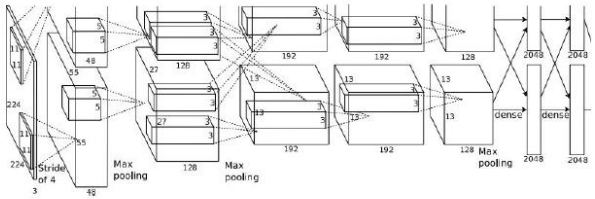
Each image needs a different number of outputs!



CAT: (x, y, w, h) — 4 numbers



DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h) — 16 numbers

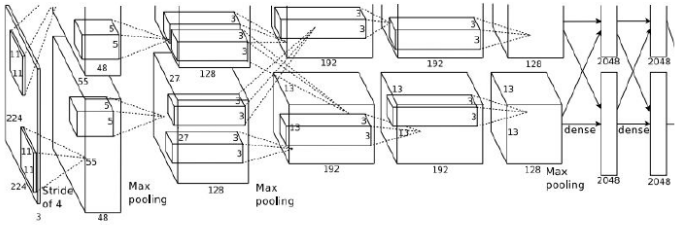


DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
.... — Many numbers!

Pristup pomerajućeg prozora

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

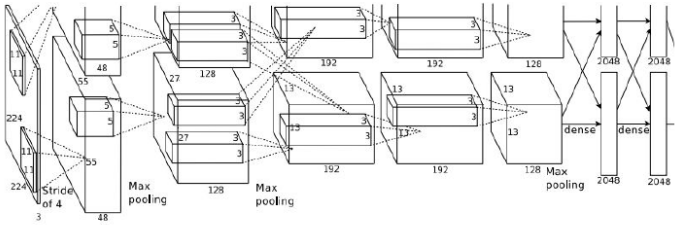


Dog? NO
Cat? NO
Background? YES

Pristup pomerajućeg prozora

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

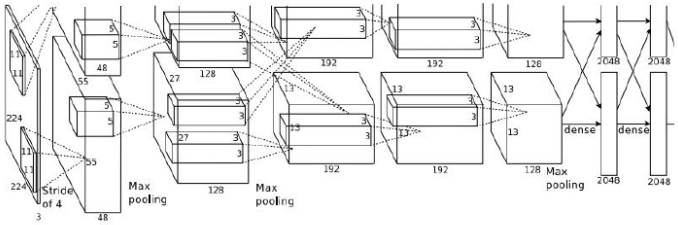


Dog? YES
Cat? NO
Background? NO

Pristup pomerajućeg prozora

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

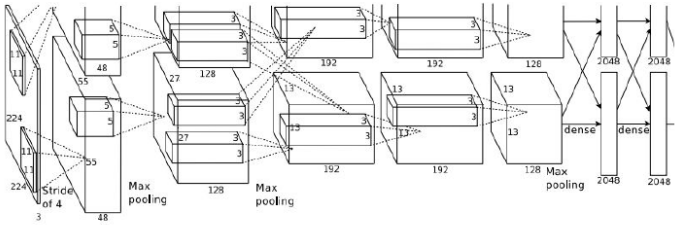


Dog? YES
Cat? NO
Background? NO

Pristup pomerajućeg prozora

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background

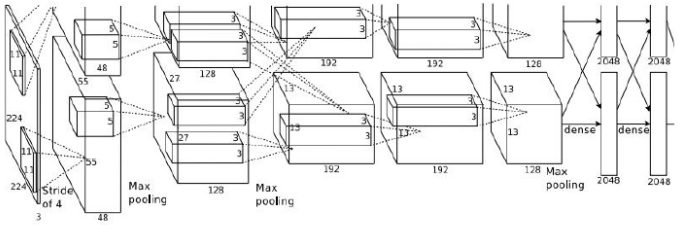


Dog? NO
Cat? YES
Background? NO

Pristup pomerajućeg prozora

Object Detection as Classification: Sliding Window

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



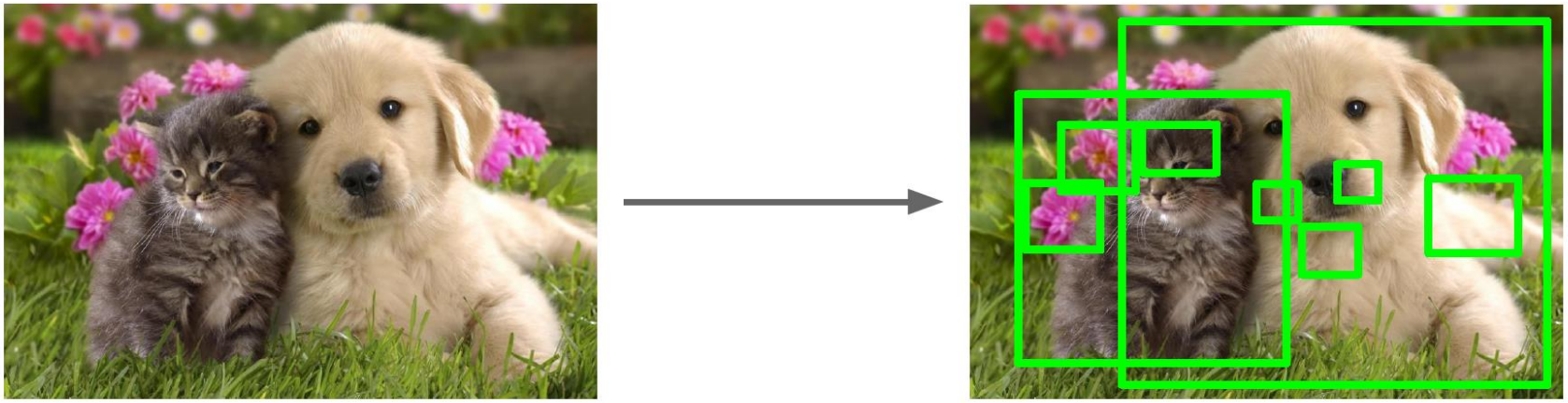
Dog? NO
Cat? YES
Background? NO

Problem: Need to apply CNN to huge number of locations and scales, very computationally expensive!

Region Proposals

Region Proposals

- Find “blobby” image regions that are likely to contain objects
- Relatively fast to run; e.g. Selective Search gives 1000 region proposals in a few seconds on CPU



Alexe et al, “Measuring the objectness of image windows”, TPAMI 2012
Uijlings et al, “Selective Search for Object Recognition”, IJCV 2013
Cheng et al, “BING: Binarized normed gradients for objectness estimation at 300fps”, CVPR 2014
Zitnick and Dollar, “Edge boxes: Locating object proposals from edges”, ECCV 2014

R-CNN (*Regions with Convolutional Neural Network Features*)

R-CNN

Input image

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN (*Regions with Convolutional Neural Network Features*)

R-CNN

Problem: ROI mogu imati različite dimenzije, a, ako treba da ih pošaljemo u CNN za klasifikaciju, treba da budu iste veličine (zbog *fully connected* slojeva)

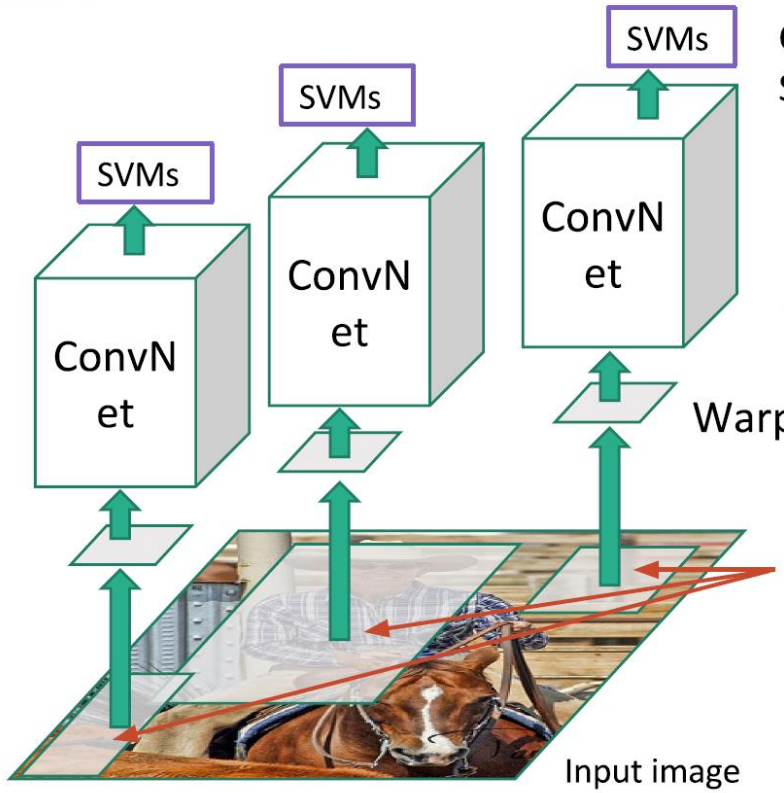
Input image

Regions of Interest (RoI) from a proposal method (~2k)

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN (*Regions with Convolutional Neural Network Features*)

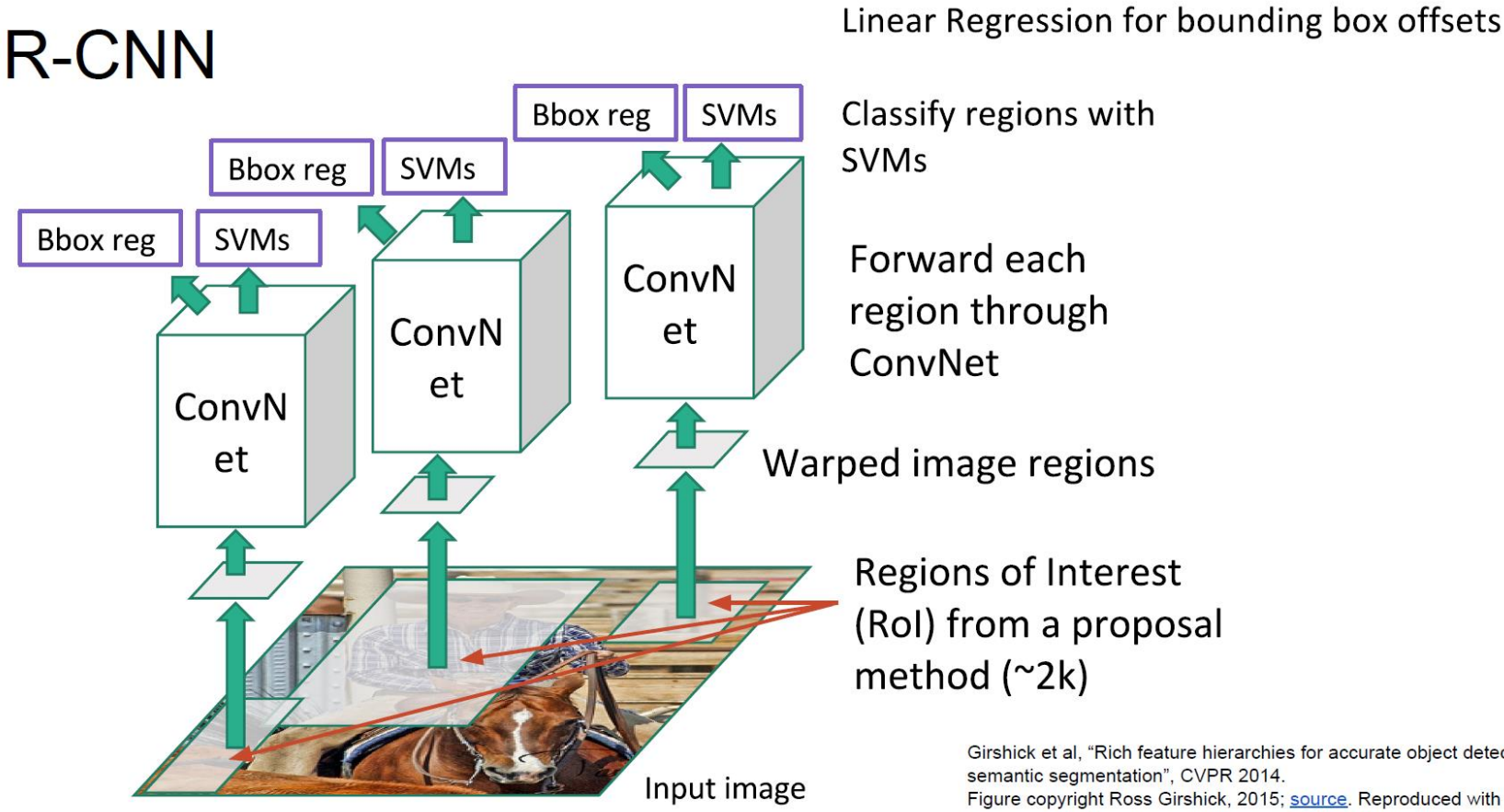
R-CNN



Konvertujemo ROI u kvadratne regije fiksne veličine

Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN (*Regions with Convolutional Neural Network Features*)

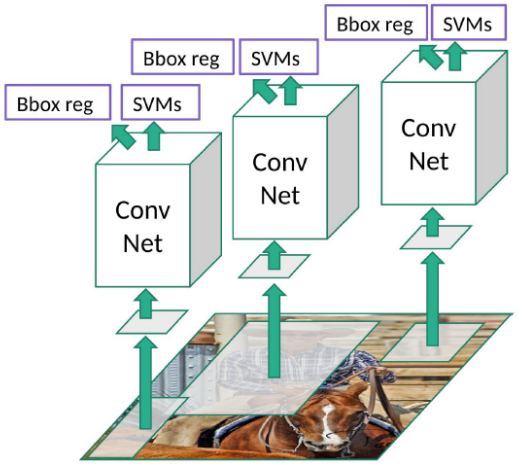


Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

R-CNN nedostaci

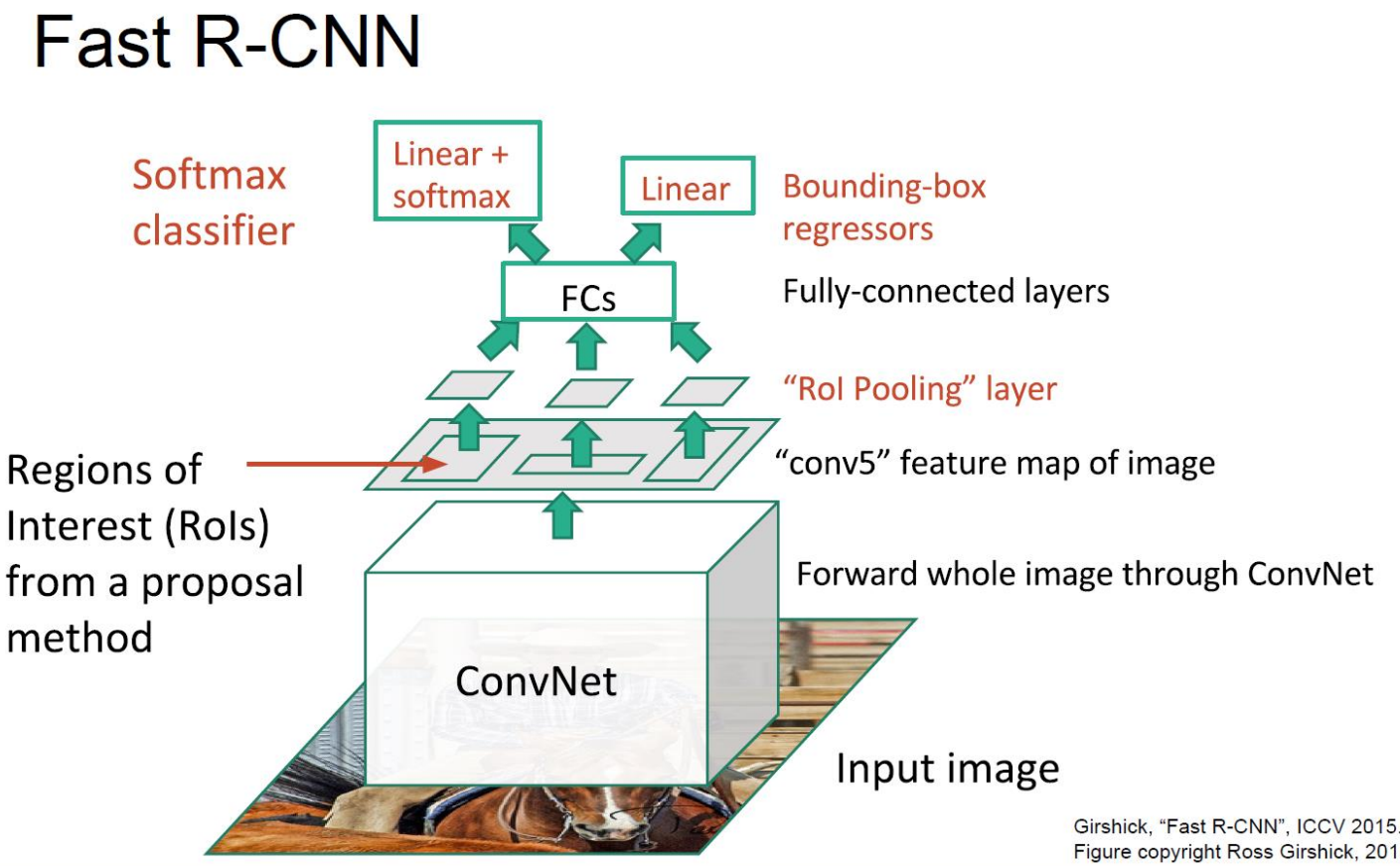
R-CNN: Problems

- Ad hoc training objectives
  - Fine-tune network with softmax classifier (log loss)
  - Train post-hoc linear SVMs (hinge loss)
  - Train post-hoc bounding-box regressions (least squares)
- Training is slow (84h), takes a lot of disk space
- Inference (detection) is slow
  - 47s / image with VGG16 [Simonyan & Zisserman. ICLR15]
  - Fixed by SPP-net [He et al. ECCV14]



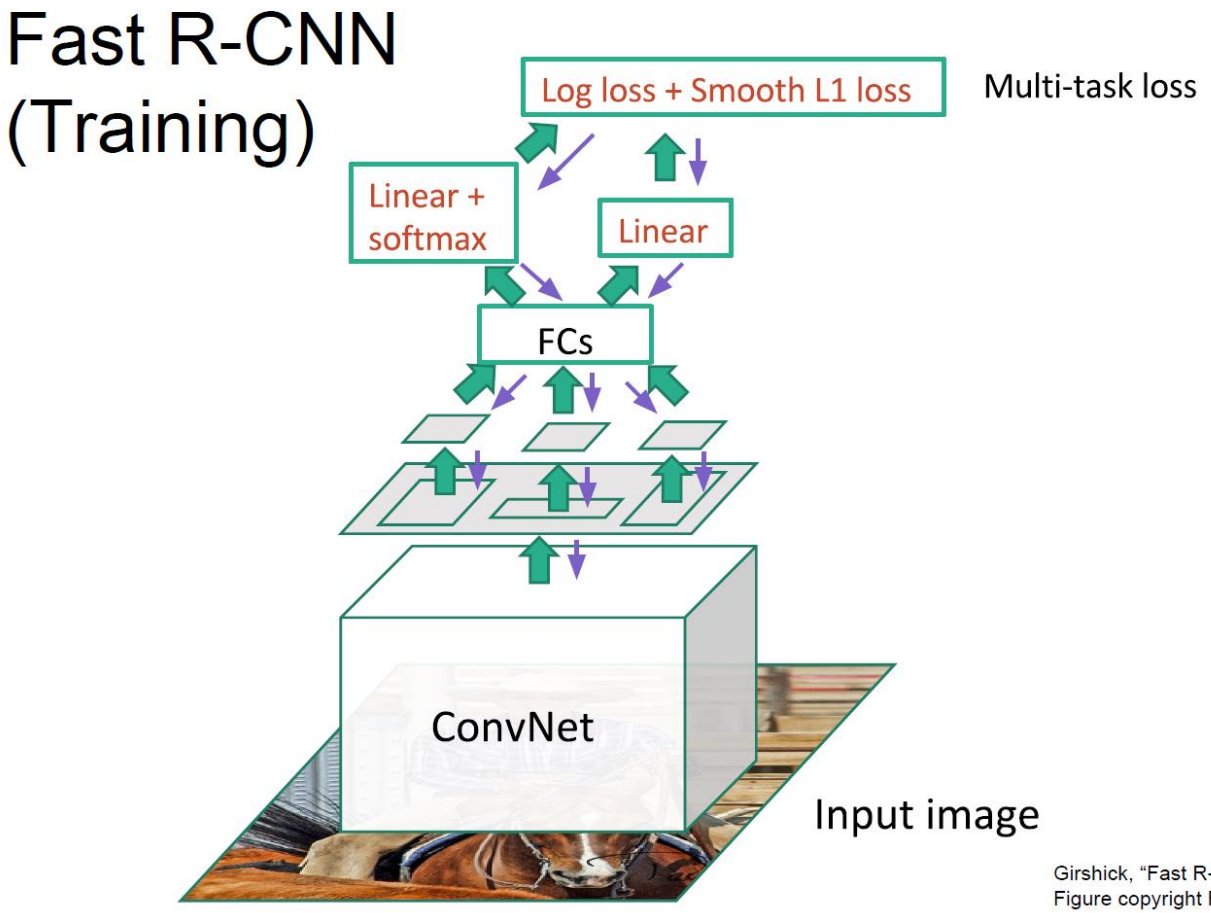
Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
Slide copyright Ross Girshick, 2015; source. Reproduced with permission.

Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

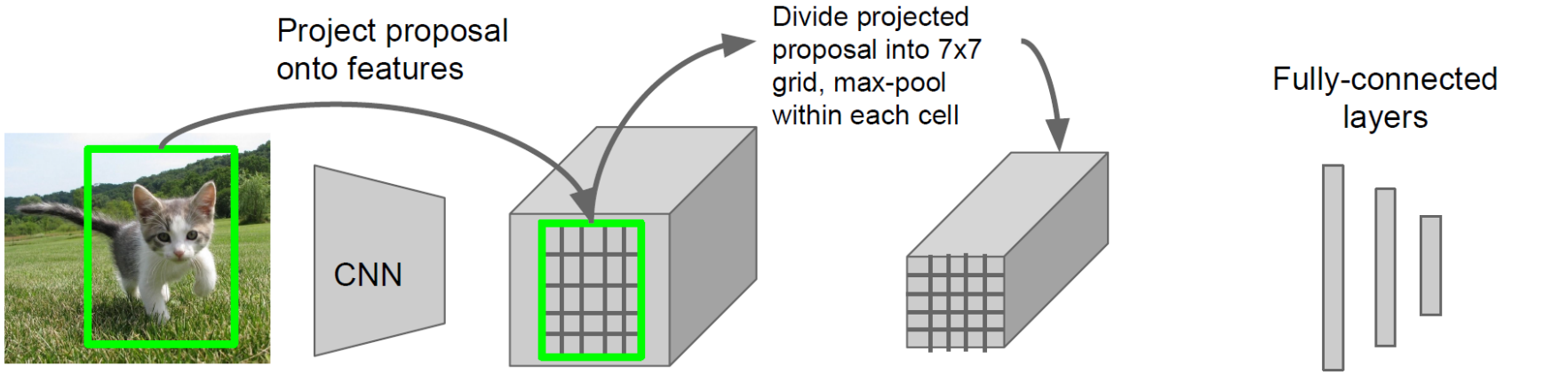
Fast R-CNN



Girshick, "Fast R-CNN", ICCV 2015.
Figure copyright Ross Girshick, 2015; source. Reproduced with permission.

RoI Pooling

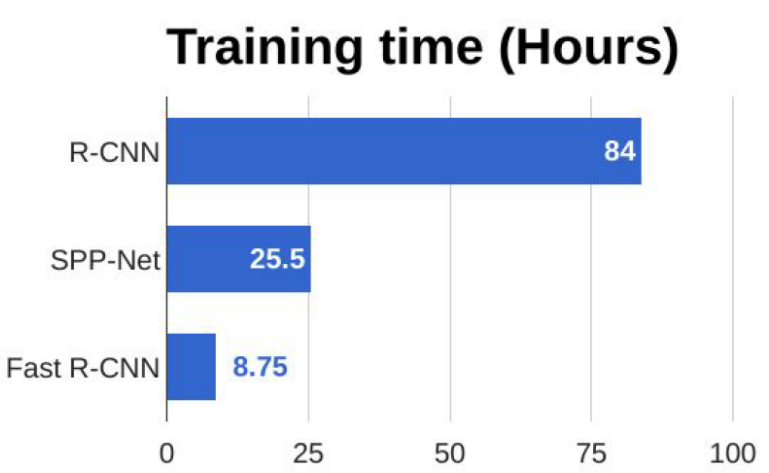
Faster R-CNN: RoI Pooling

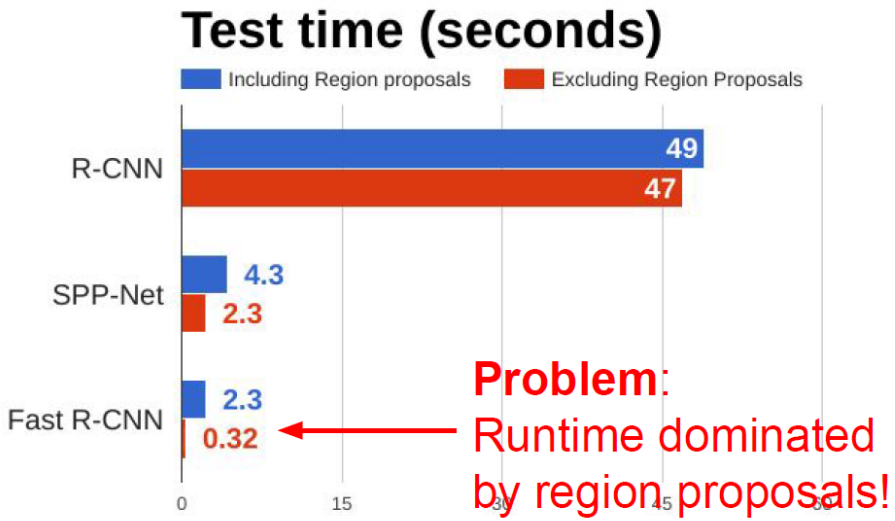


Girshick, "Fast R-CNN", ICCV 2015.

Poređenje R-CNN i Fast R-CNN

R-CNN vs SPP vs Fast R-CNN





Girshick et al, "Rich feature hierarchies for accurate object detection and semantic segmentation", CVPR 2014.
He et al, "Spatial pyramid pooling in deep convolutional networks for visual recognition", ECCV 2014
Girshick, "Fast R-CNN", ICCV 2015

Faster R-CNN

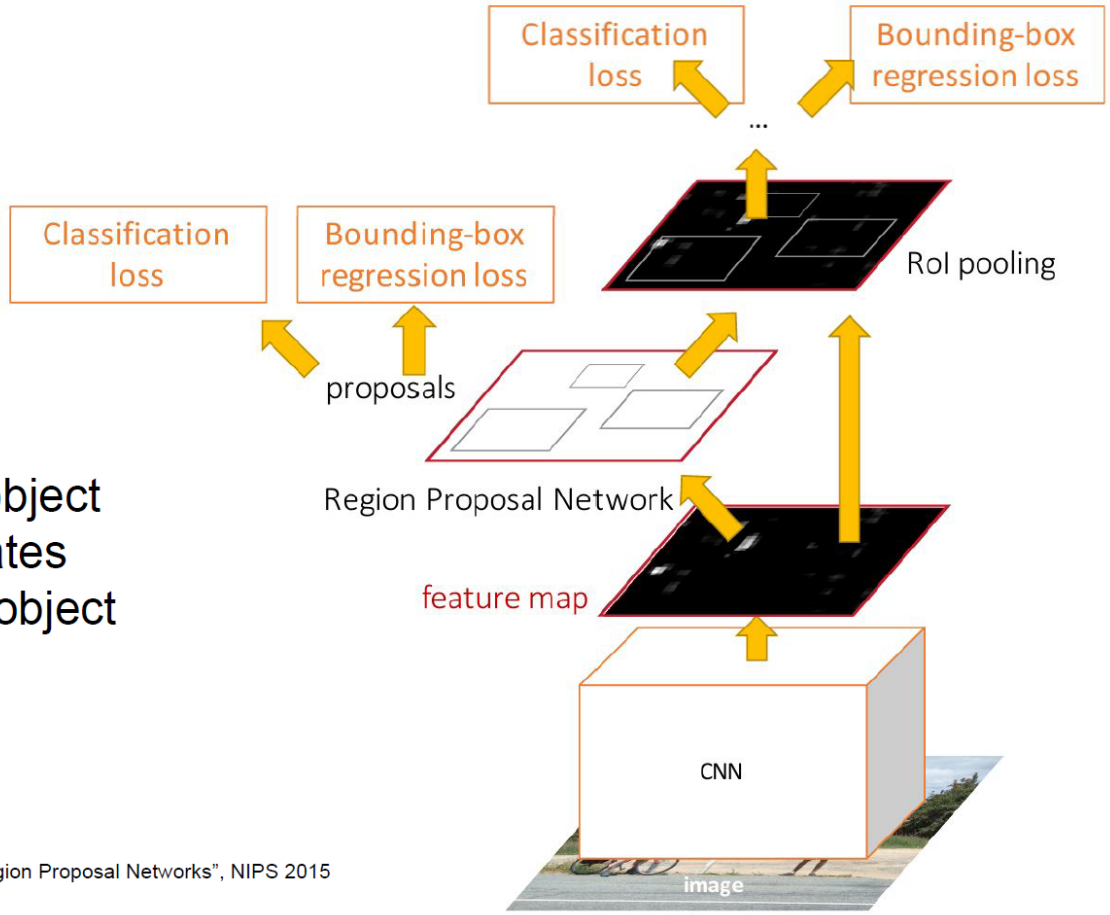
Faster R-CNN:

Make CNN do proposals!

Insert **Region Proposal Network (RPN)** to predict proposals from features

Jointly train with 4 losses:

1. RPN classify object / not object
2. RPN regress box coordinates
3. Final classification score (object classes)
4. Final box coordinates

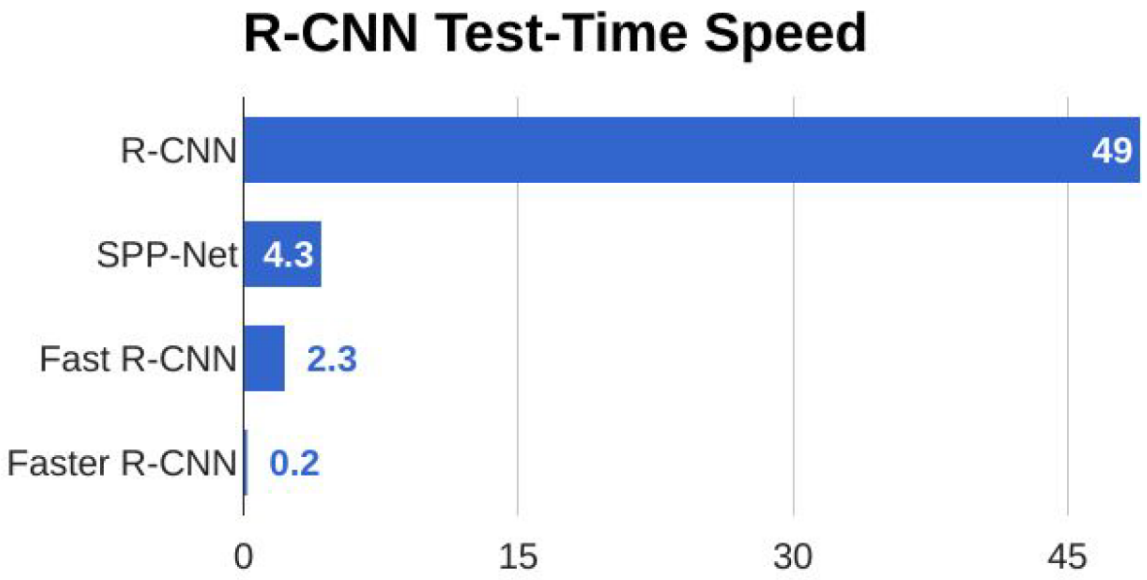


Ren et al, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", NIPS 2015
Figure copyright 2015, Ross Girshick; reproduced with permission

Faster R-CNN

Faster R-CNN:

Make CNN do proposals!

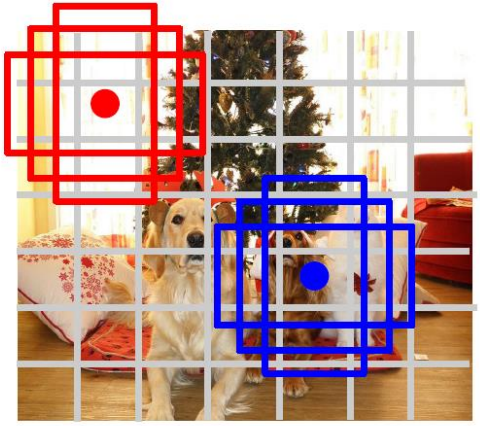


Bez predlaganja regija

Detection without Proposals: YOLO / SSD

Go from input image to tensor of scores with one big convolutional network!

Input image
3 x H x W



Divide image into grid
7 x 7

Image a set of **base boxes** centered at each grid cell
Here B = 3

Within each grid cell:

- Regress from each of the B base boxes to a final box with 5 numbers: (dx, dy, dh, dw, confidence)
- Predict scores for each of C classes (including background as a class)

Output:
7 x 7 x (5 * B + C)

Redmon et al, "You Only Look Once: Unified, Real-Time Object Detection", CVPR 2016
Liu et al, "SSD: Single-Shot MultiBox Detector", ECCV 2016

Parametri

Object Detection: Lots of variables ...

Base Network
VGG16
ResNet-101
Inception V2
Inception V3
Inception
ResNet
MobileNet

Object Detection architecture
Faster R-CNN
R-FCN
SSD

Image Size
 # Region Proposals
...

Takeaways
Faster R-CNN is slower but more accurate

SSD is much faster but not as accurate

Huang et al, "Speed/accuracy trade-offs for modern convolutional object detectors", CVPR 2017

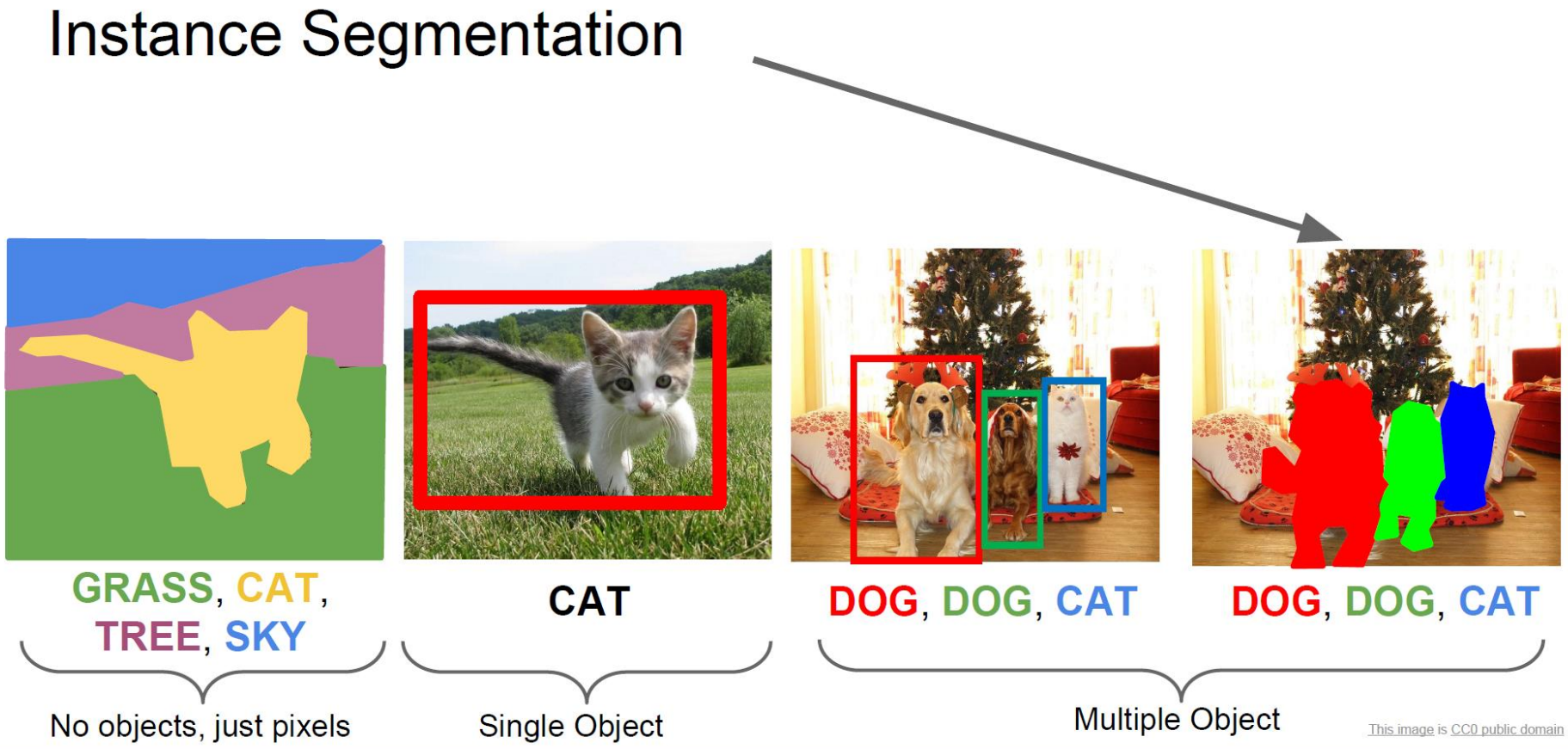
R-FCN: Dai et al, "R-FCN: Object Detection via Region-based Fully Convolutional Networks", NIPS 2016
Inception-V2: Ioffe and Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", ICML 2015
Inception V3: Szegedy et al, "Rethinking the Inception Architecture for Computer Vision", arXiv 2016
Inception ResNet: Szegedy et al, "Inception-V4, Inception-ResNet and the Impact of Residual Connections on Learning", arXiv 2016
MobileNet: Howard et al, "Efficient Convolutional Neural Networks for Mobile Vision Applications", arXiv 2017

Molba za učešće u istraživanju

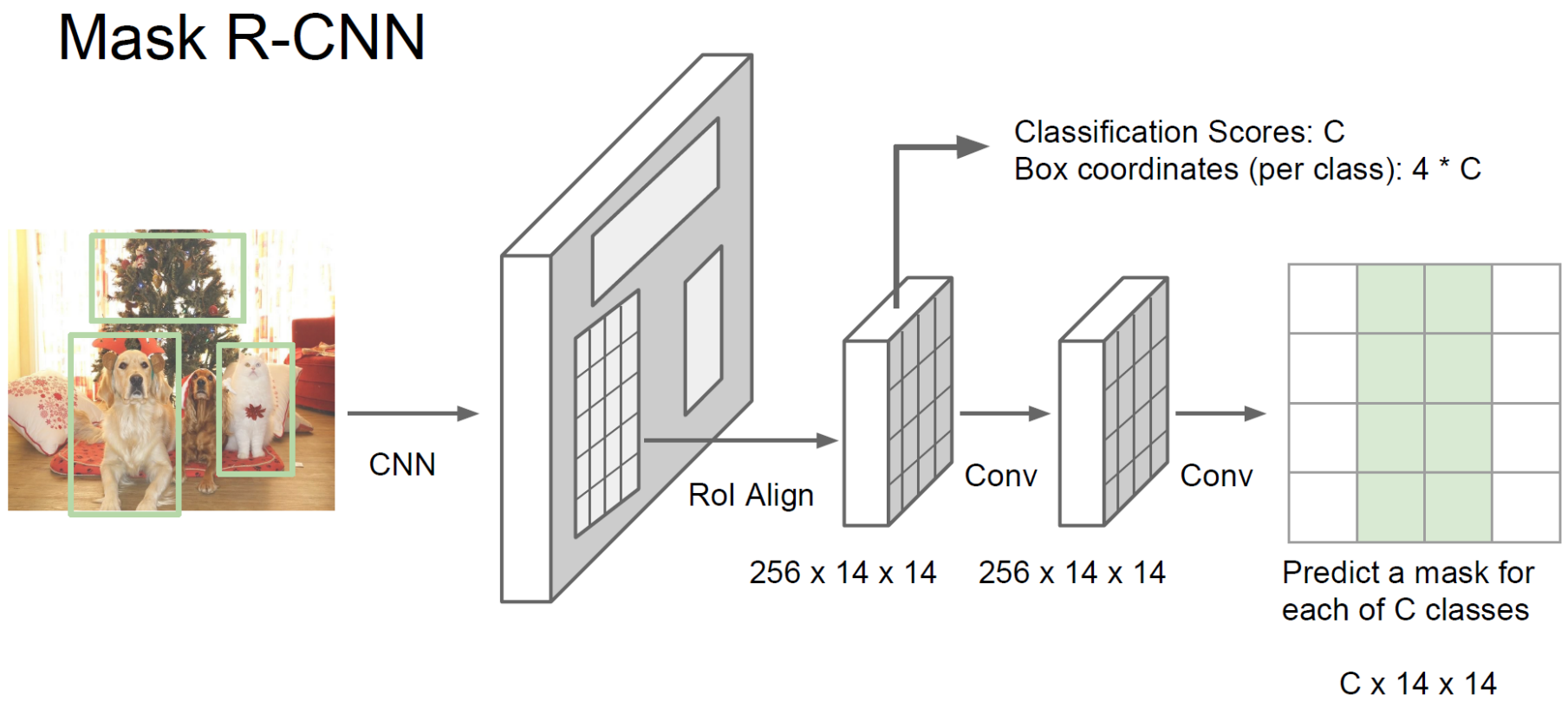
- Predikcija čitljivosti koda
  - Python snippeti koje treba da ocenite iz aspekta čitljivosti
  - Praćenje očiju *eye-tracker*-om kako bismo videli tačno koji delovi koda se dublje analiziraju

- Učešće
  - Poznavanje *Python* jezika
  - Forma za prijavu: https://docs.google.com/forms/d/e/1FAIpQLScnnxxCJgKZWWIXSDDlG3-me2i9_pJkDjnDbqKDAid8Fwuoow/viewform
  - Rok: sreda (5.12.) uveče

Instance segmentation



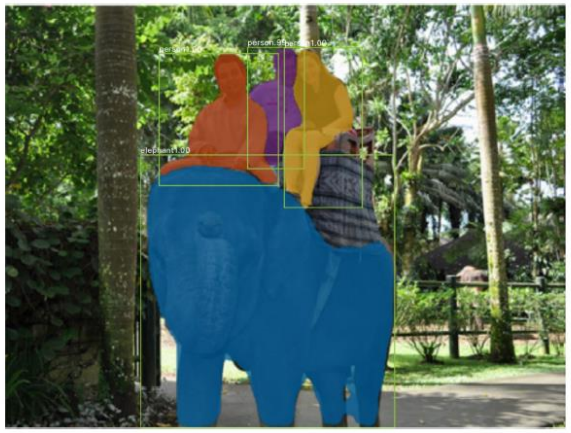
Instance segmentation state-of-the-art

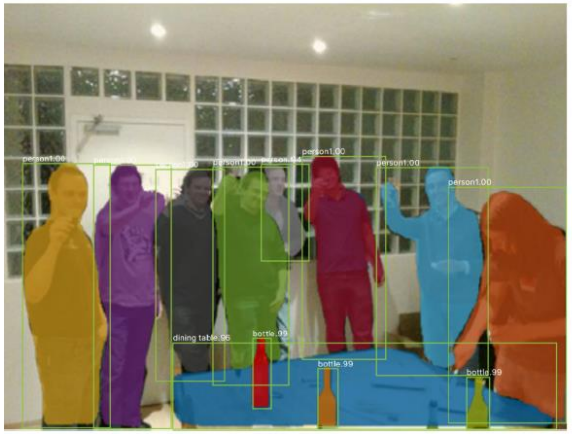


He et al, "Mask R-CNN", arXiv 2017

Instance segmentation state-of-the-art

Mask R-CNN: Very Good Results!

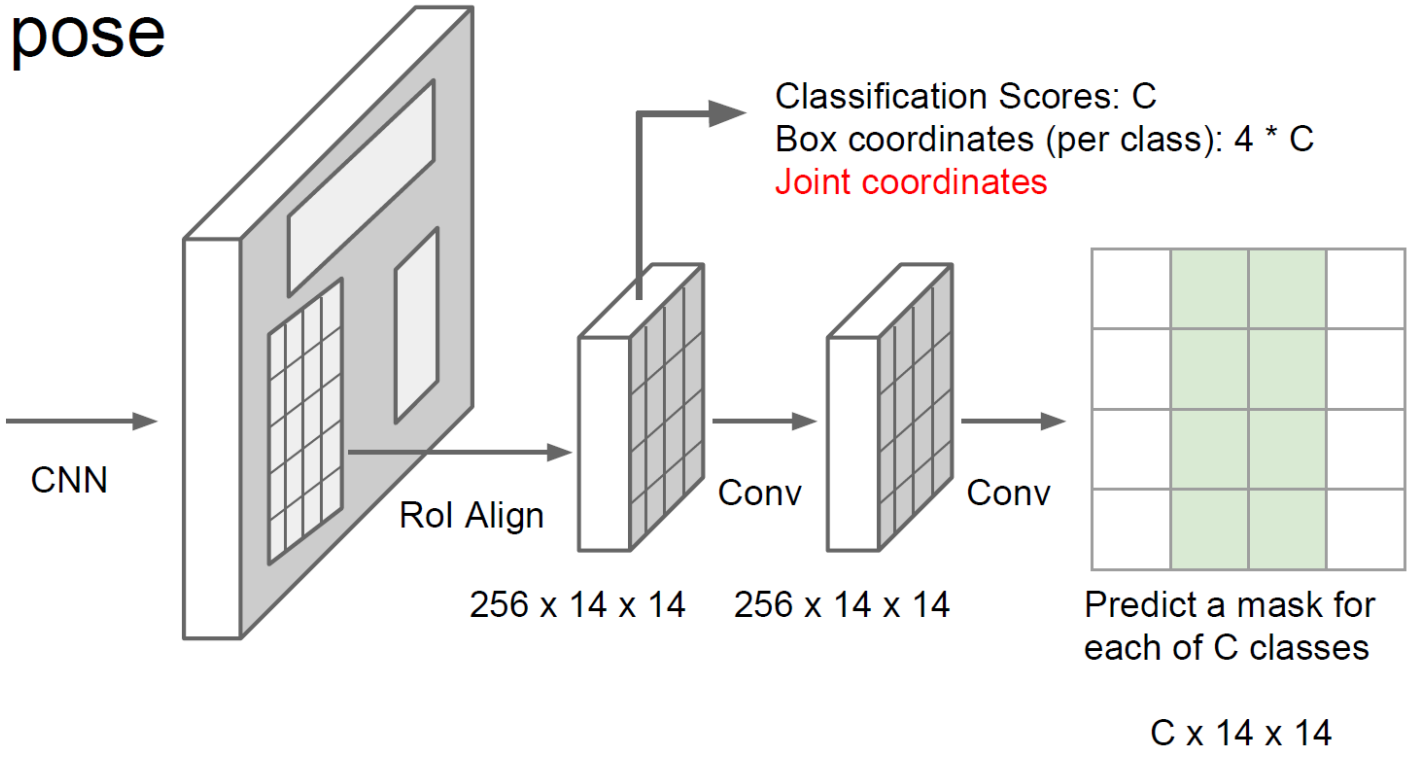




He et al, "Mask R-CNN", arXiv 2017
Figures copyright Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, 2017.
Reproduced with permission.

Instance segmentation state-of-the-art

Mask R-CNN
Also does pose



He et al, "Mask R-CNN", arXiv 2017

Instance segmentation state-of-the-art

Mask R-CNN
Also does pose

He et al, "Mask R-CNN", arXiv 2017
Figures copyright Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick, 2017.
Reproduced with permission.