Deep Learning

Episode 7

Advanced computer vision

Yandex
Data Factory

LAMBDA

British Hedgehog
Preservation Society

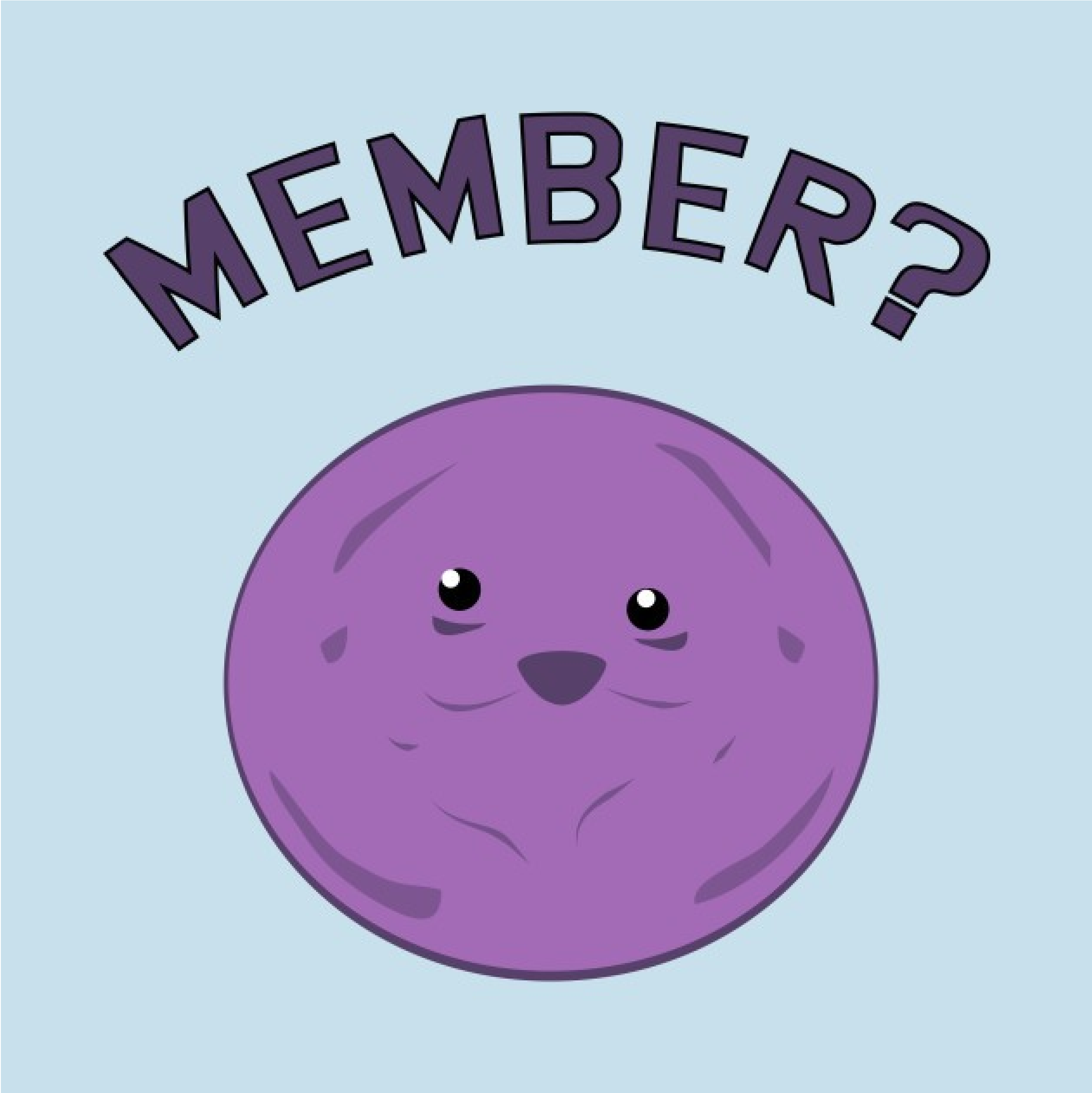

MEMBER?

Image recognition

“Dog”

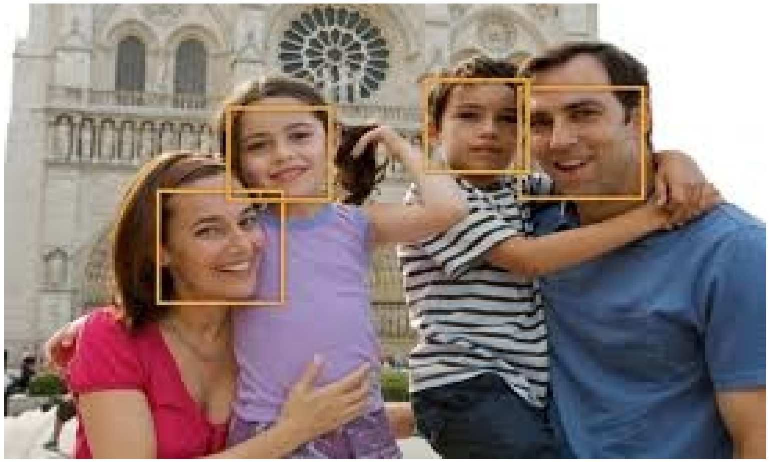
Bounding box regression

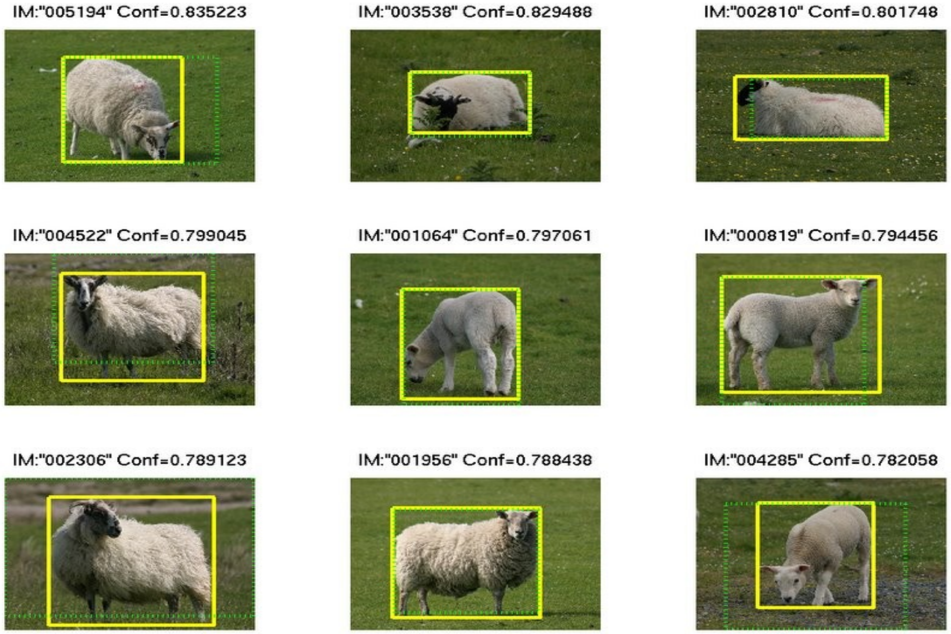
Predict object bounding box

(x0,y0,w,h)

or several bounding boxes for multiple objects.

Applications examples:
- Face detection @ cameras
- Surveillance cameras
- Self-driving cars

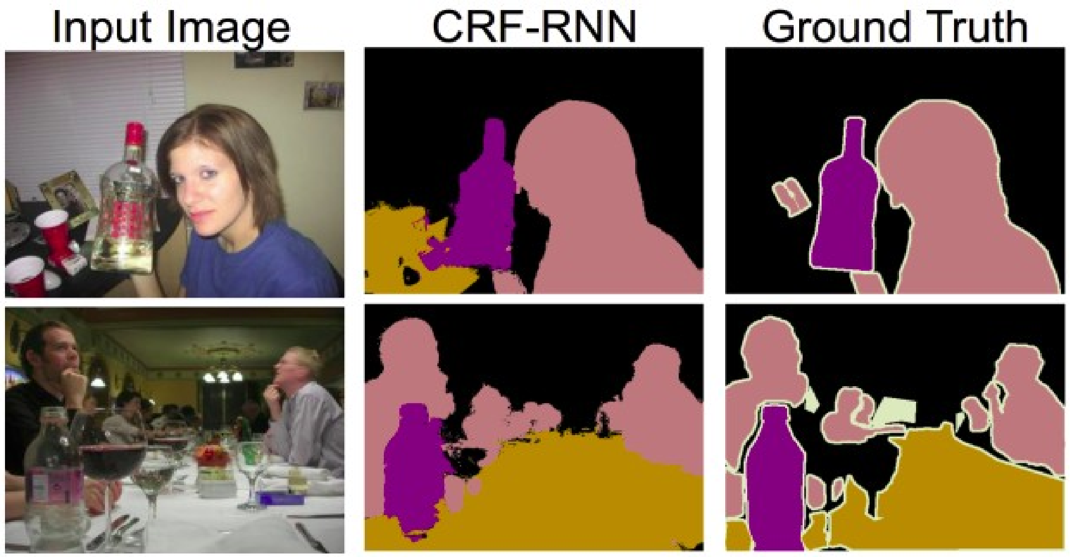




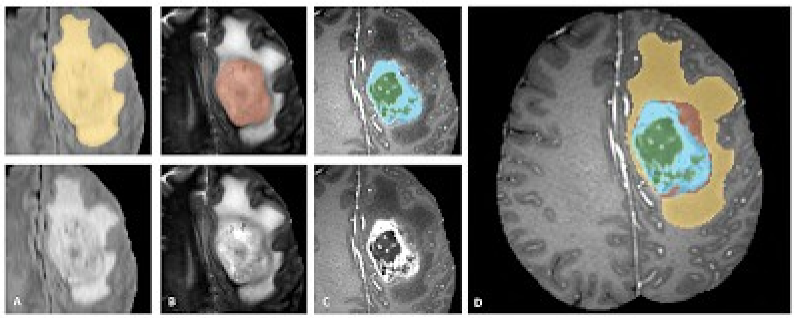
Segmentation

Predict class for each pixel

(fully-convolutional networks)



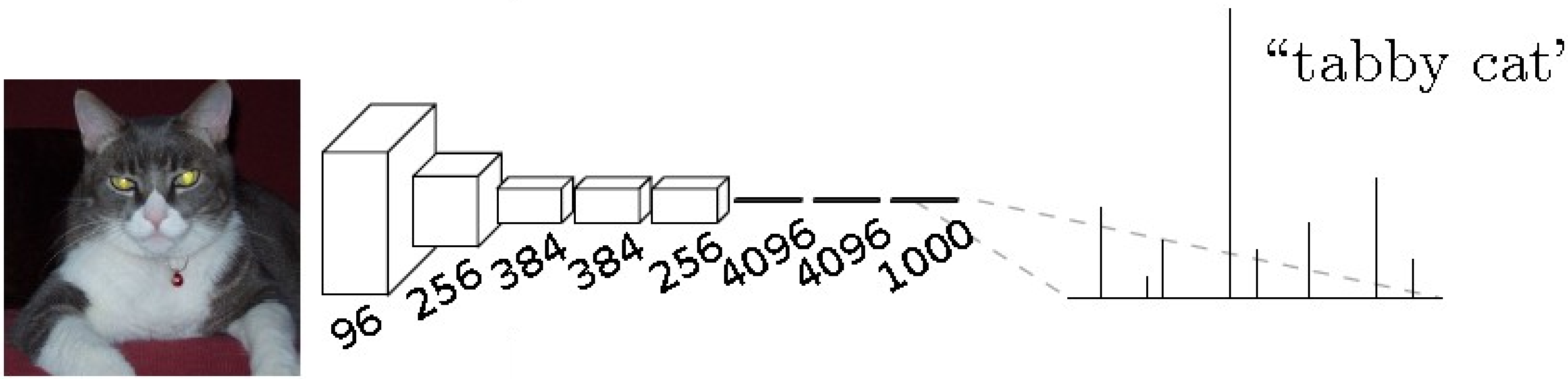
Applications examples:
- Moar surveillance
- Brain scan labeling
- Map labeling




MEMBER WE HAD 10+ WEEKS UNTIL DEADLINE?
OH, I 'MEMBER
imgflip.com

Semantic segmentation

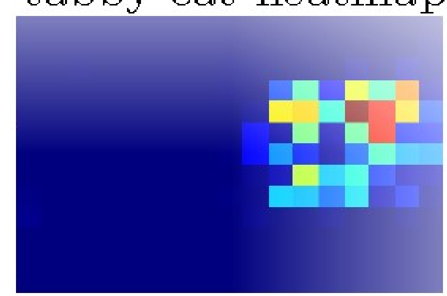
- Classification



- Segmentation/Detection

???
Ideas on how to predict that?

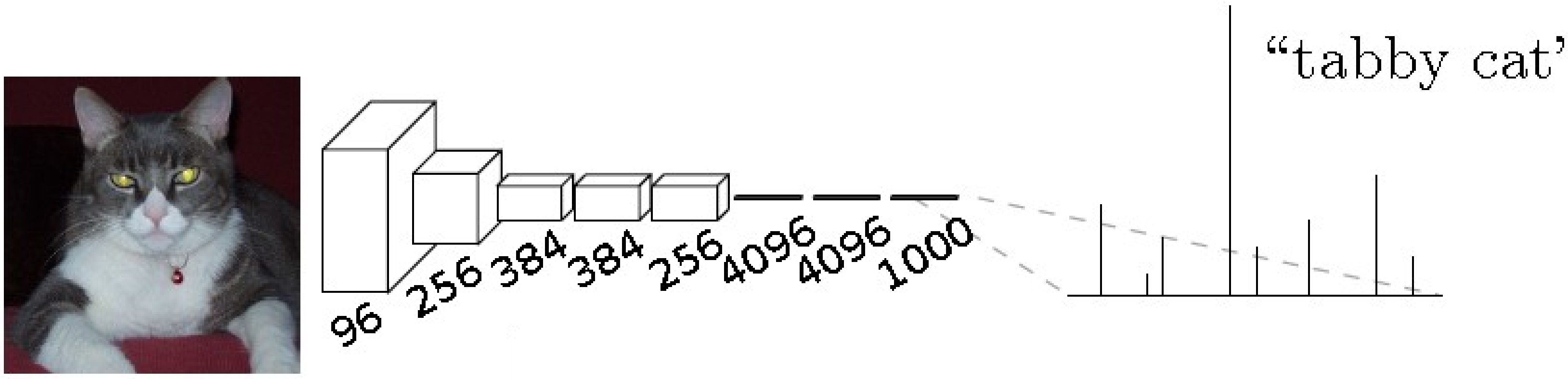
tabby cat heatmap



- Pics: V. Lempitsky

Semantic segmentation

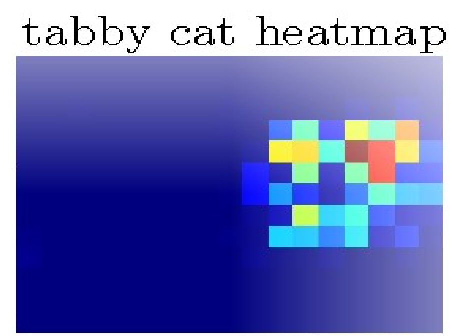
- Classification



- Segmentation/Detection (naïve)

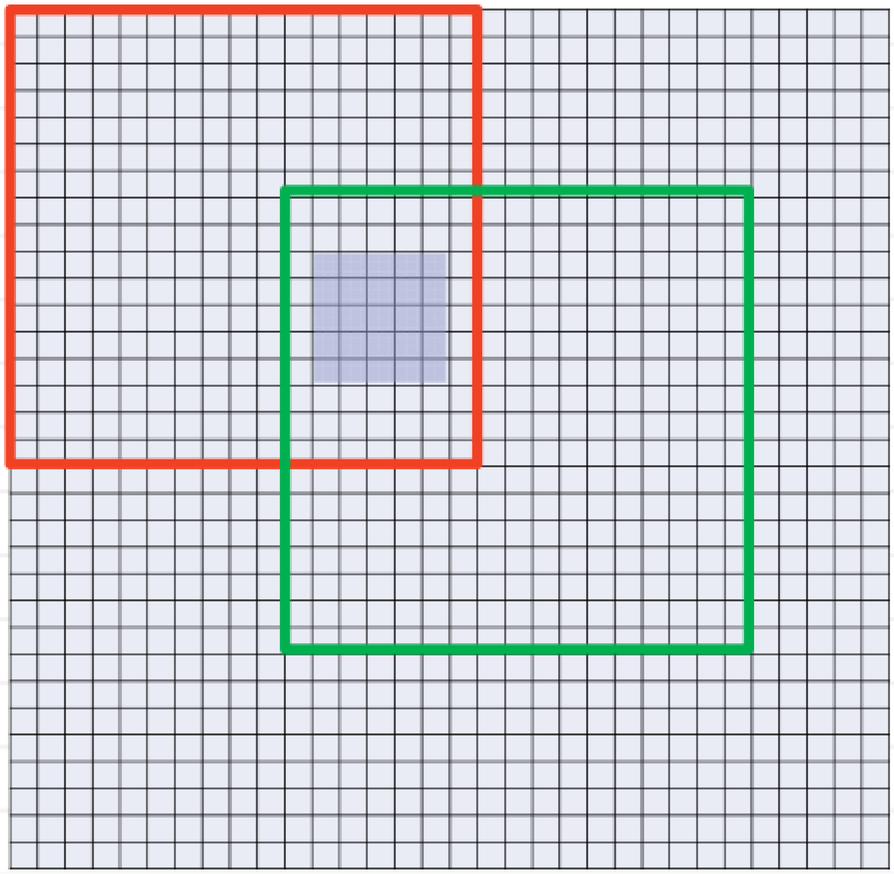
Apply to each spot

“convolution with whole NN as a filter”



- Pics: V. Lempitsky

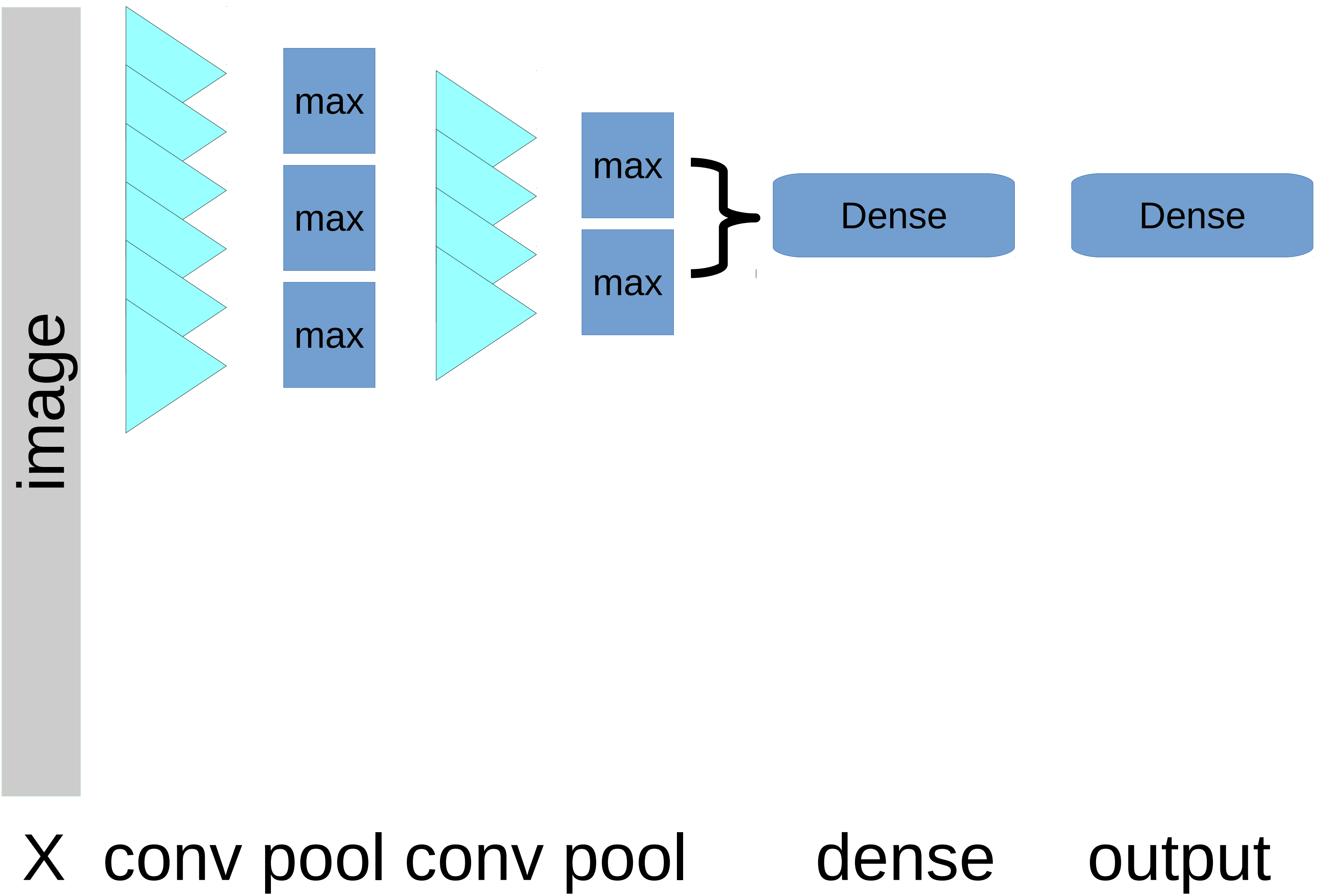
Convolutionalization



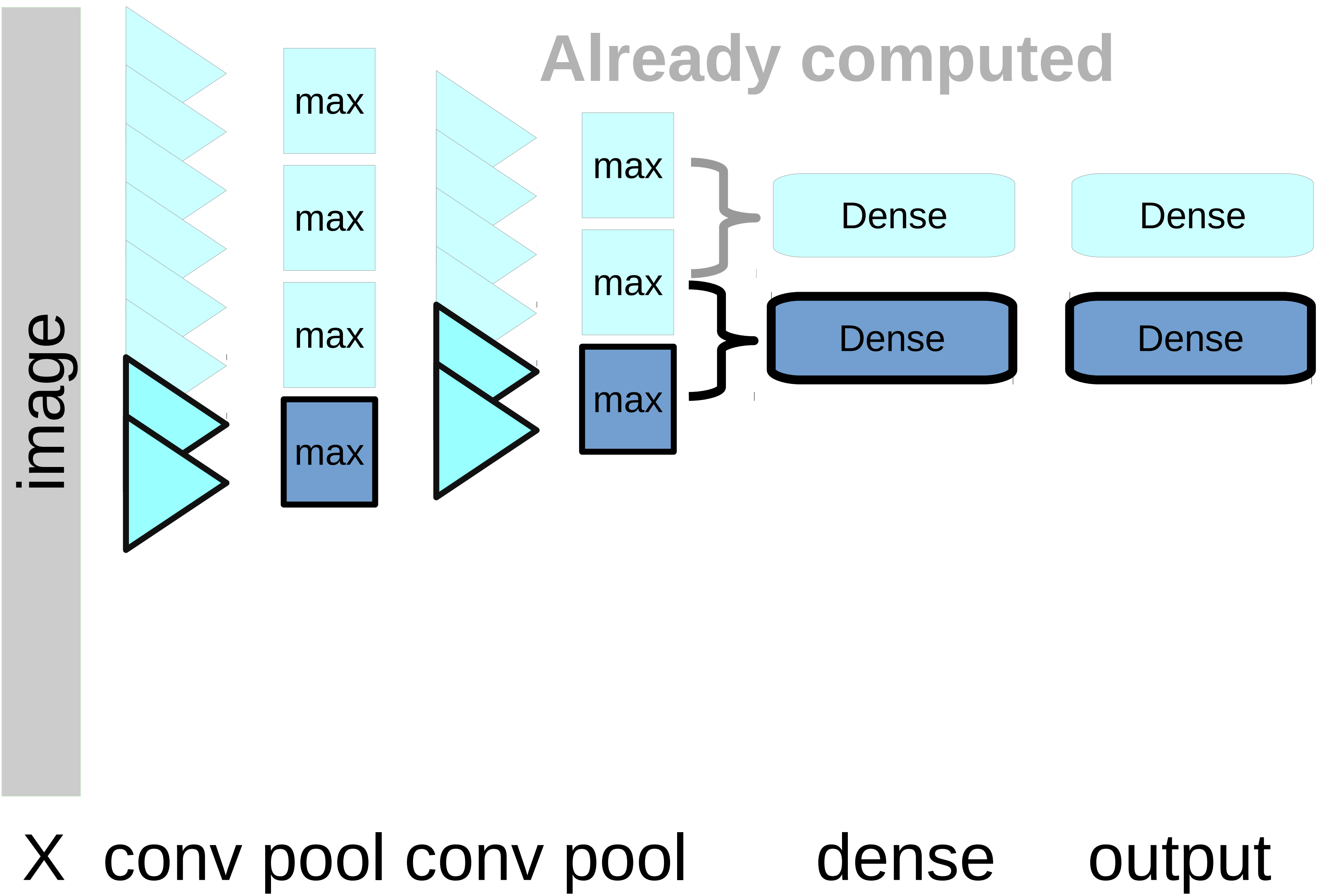
- Idea: if you apply the NN to neighboring patches, you'll have to compute same convolutions again
- Instead let us precompute convolutions for the whole image in advance.

- Pics: V. Lempitsky

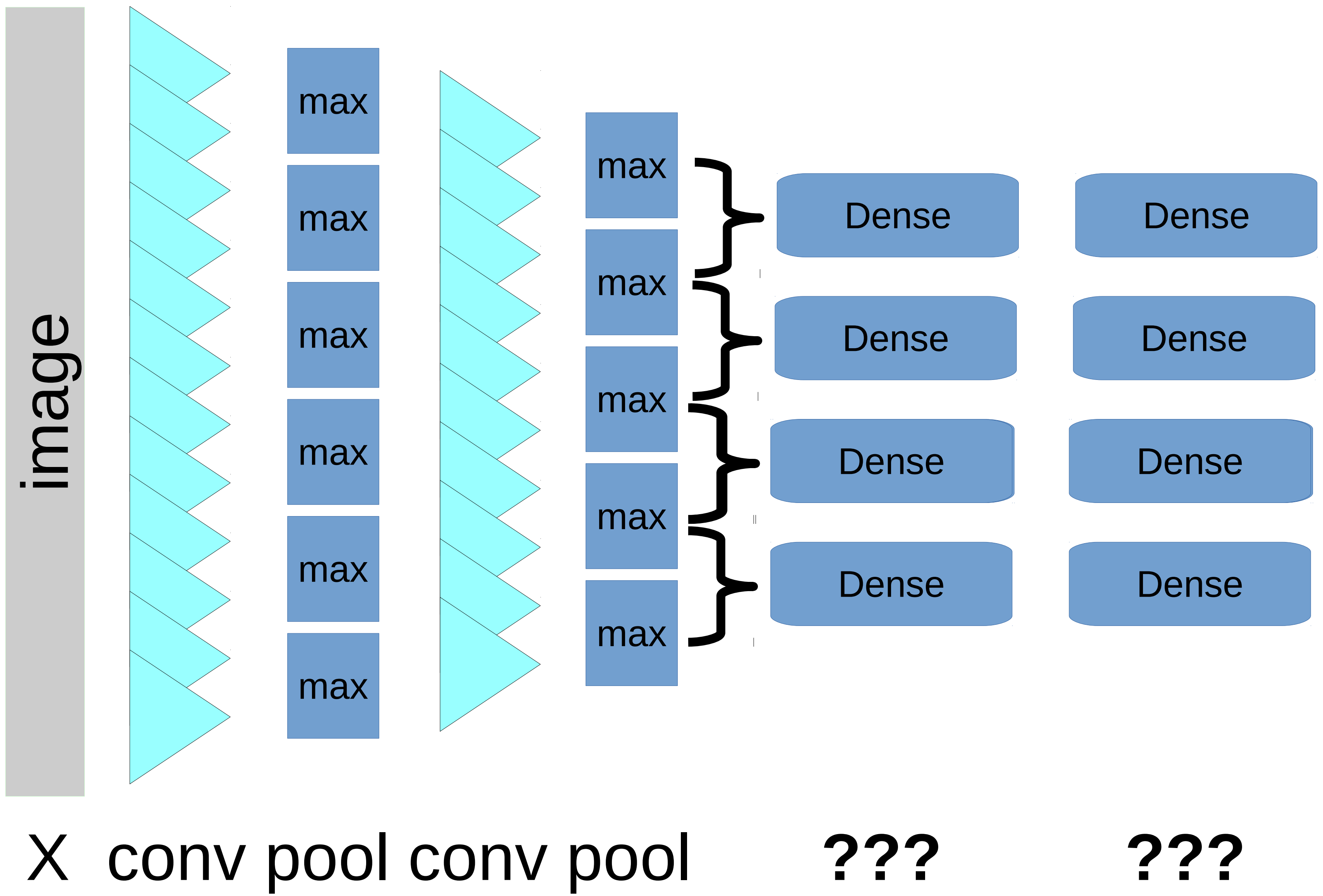
Convolutionalization



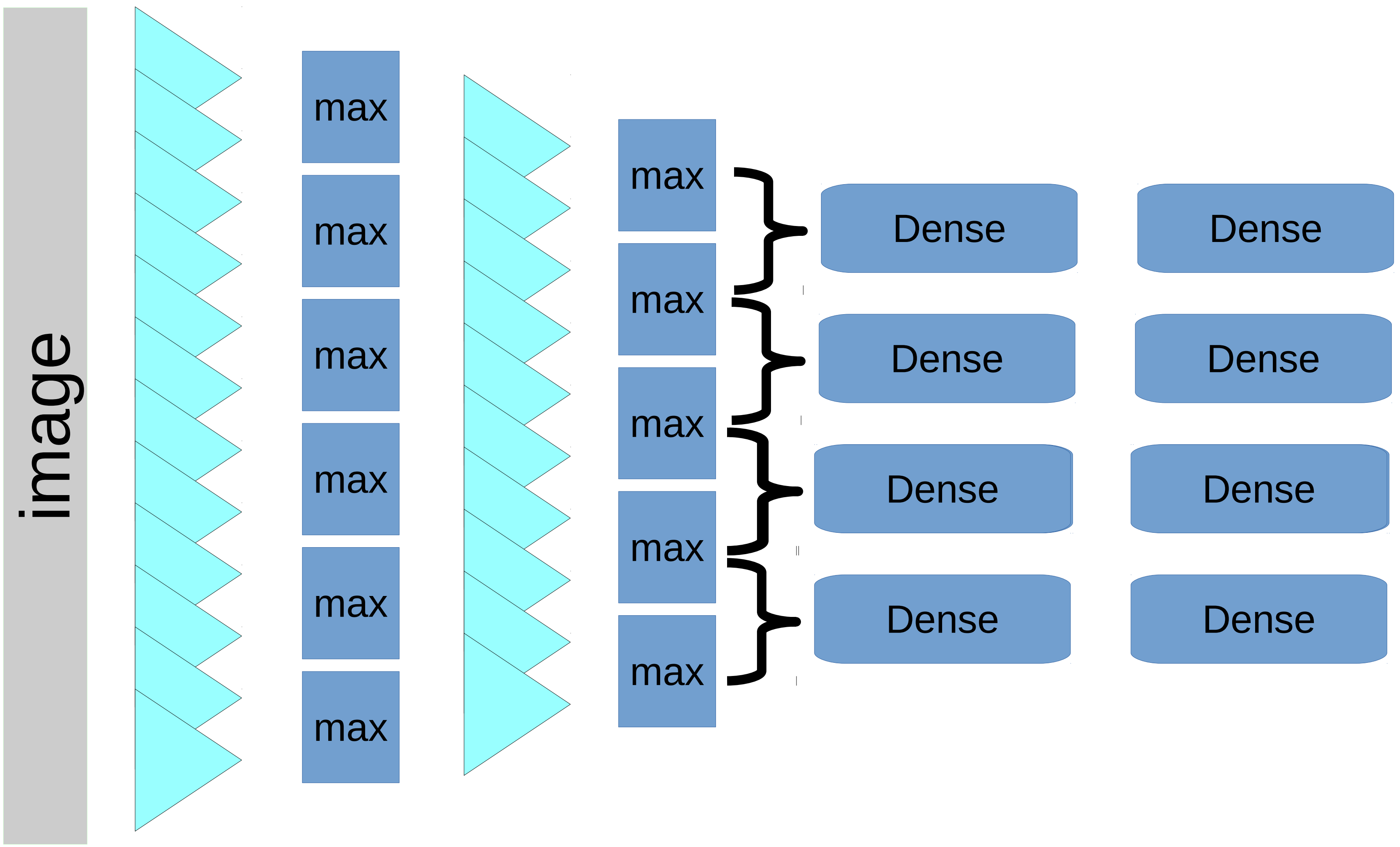
Convolutionalization



Convolutionalization



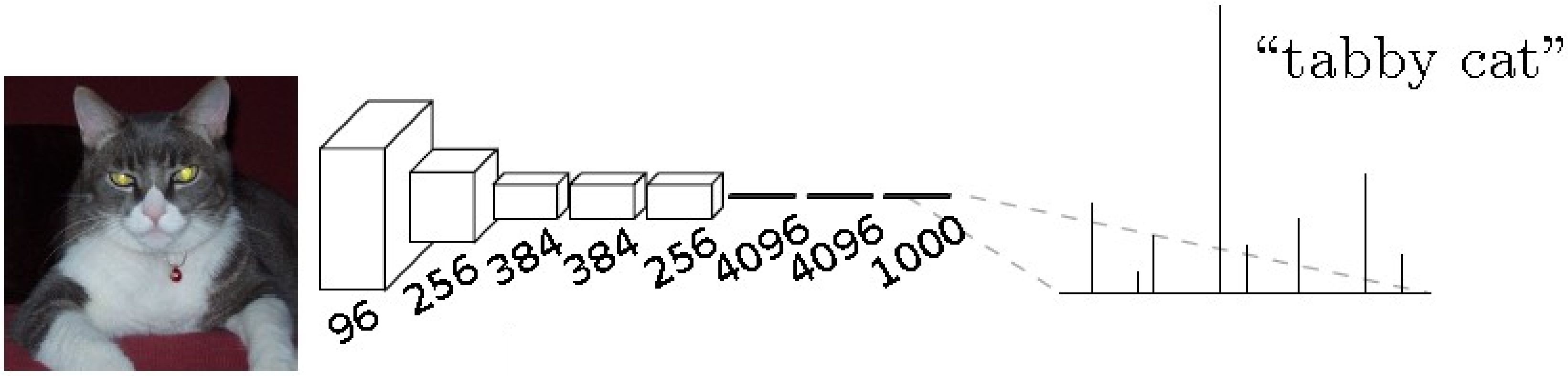
Convolutionalization



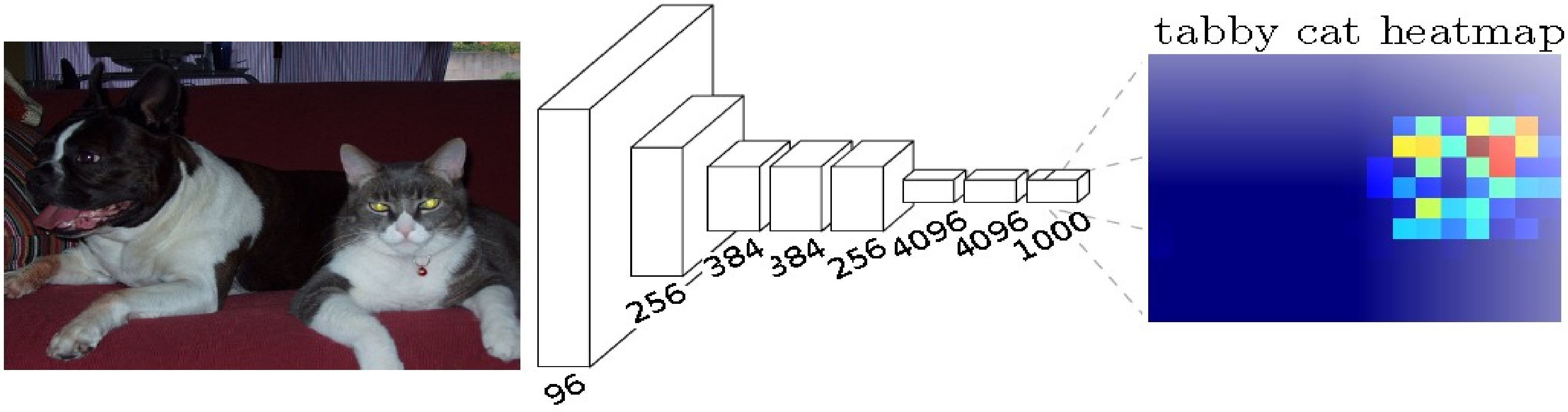
X conv pool conv pool conv2x2 conv1x1

Semantic segmentation

- Classification



- Segmentation/Detection



- Pics: V. Lempitsky

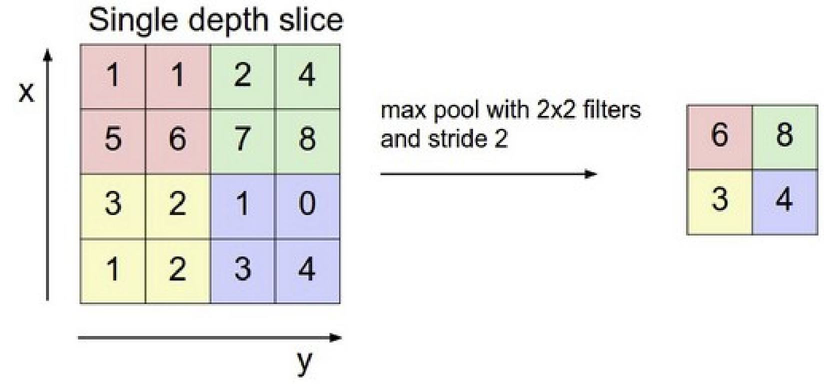
What if output must have same resolution as in the input image? Or even bigger?

How can we **increase** activation size?

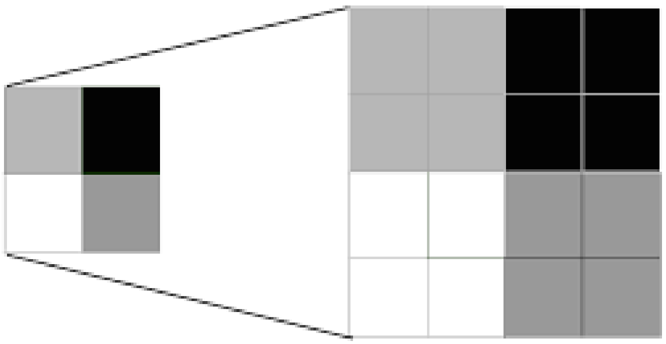
Ideas?

Upscale

- Pooling =
  - take (n*m) region,
  - output one / channel



- Upscaling / Unpooling / Upsampling
  - Take one value
  - Output (n*m) region
  - Several strategies

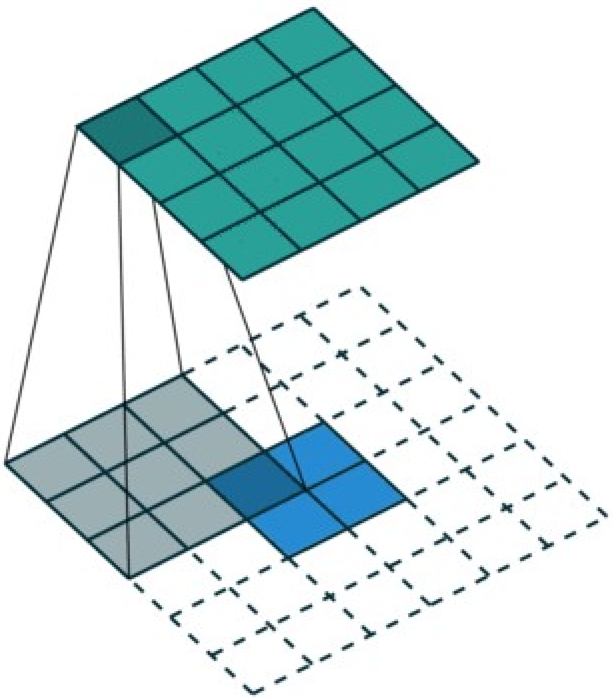


Deconvolution

- The “inverse” of convolution

  = transposed convolution

  ~ wide convolution
- deconv(conv(img))
- preserves image shape
- Stride ~ upscale



deconvolution

https://github.com/vdumoulin/conv_arithmetic

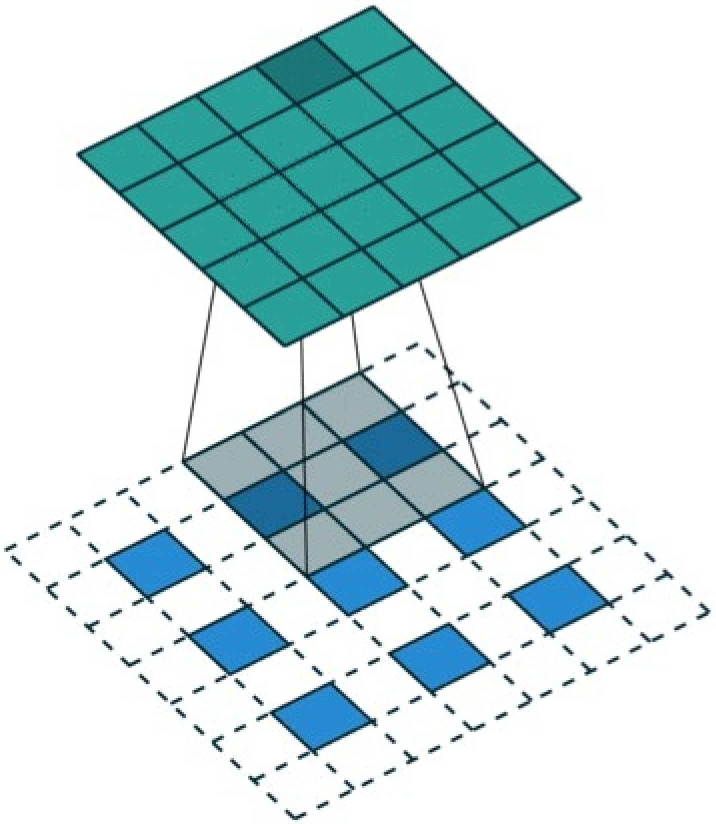
Deconvolution

- The "inverse" of convolution

  = transposed convolution

  ~ wide convolution

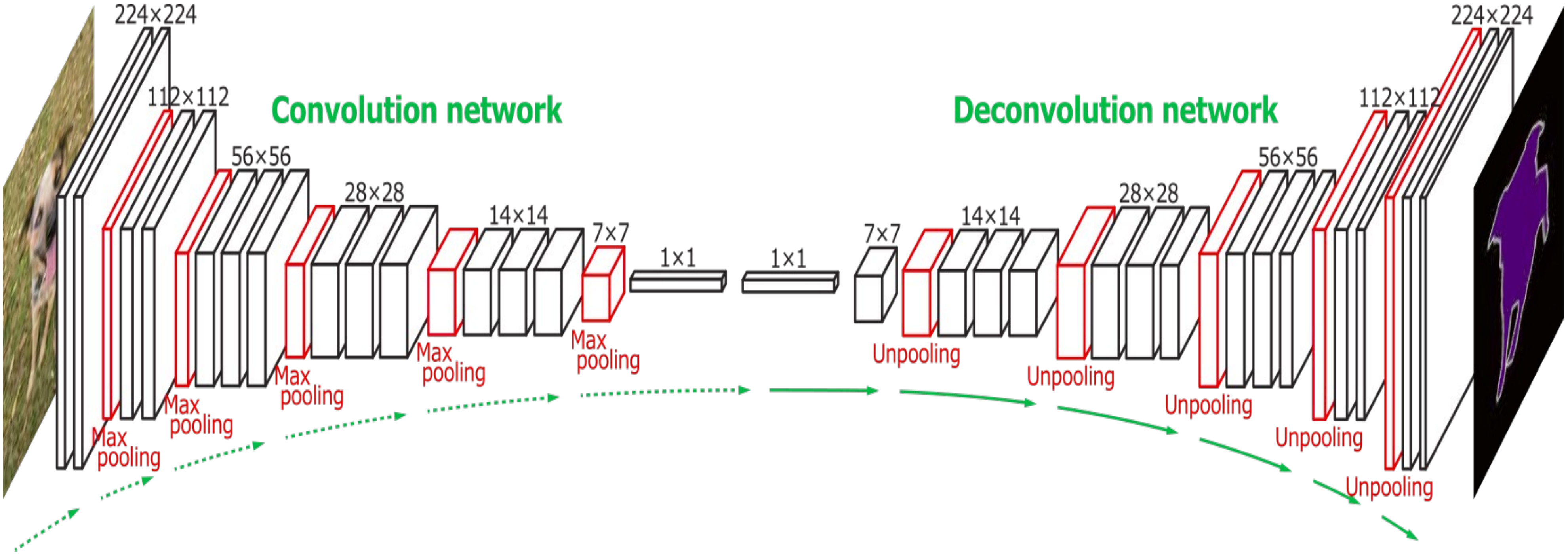
- deconv(conv(img))
- preserves image shape
- Stride ~ upscale



strided deconvolution

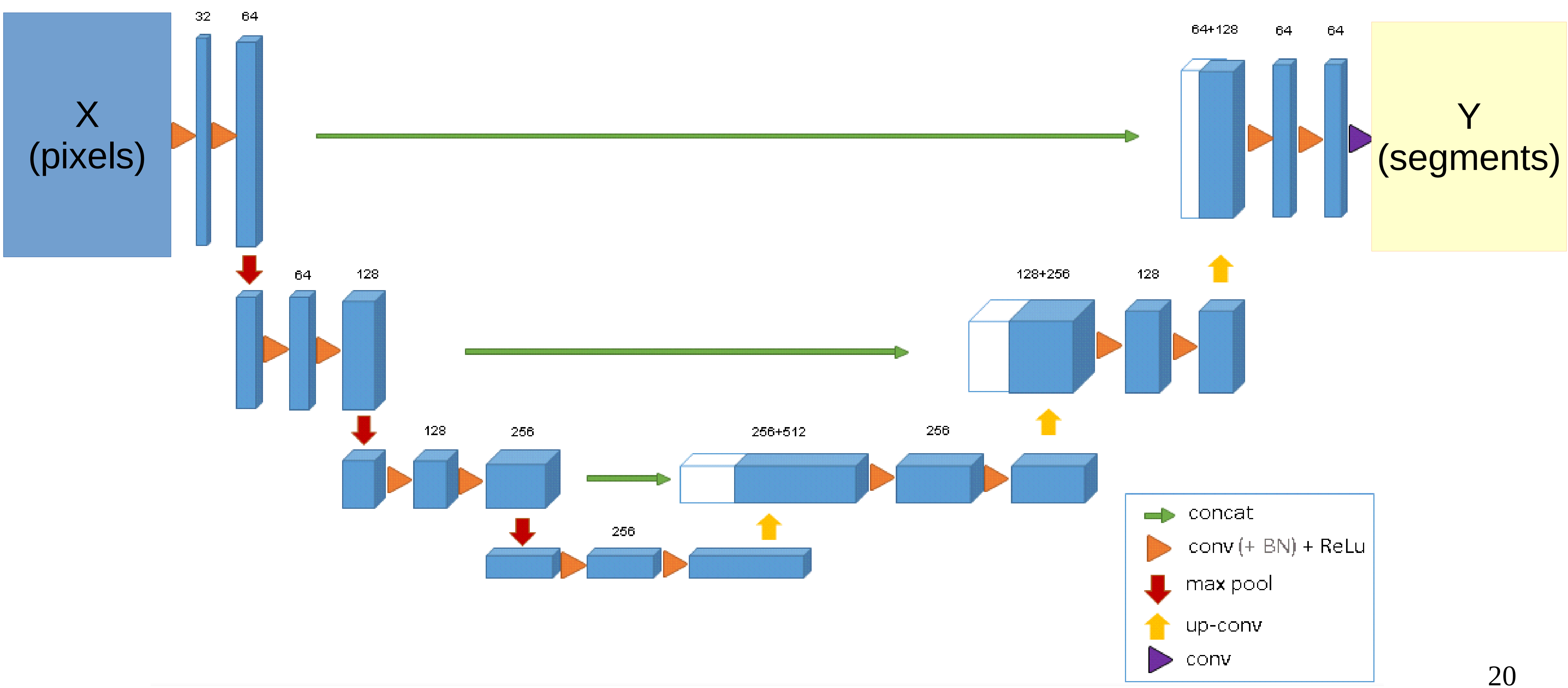
https://github.com/vdumoulin/conv_arithmetic

Fully-convolutional

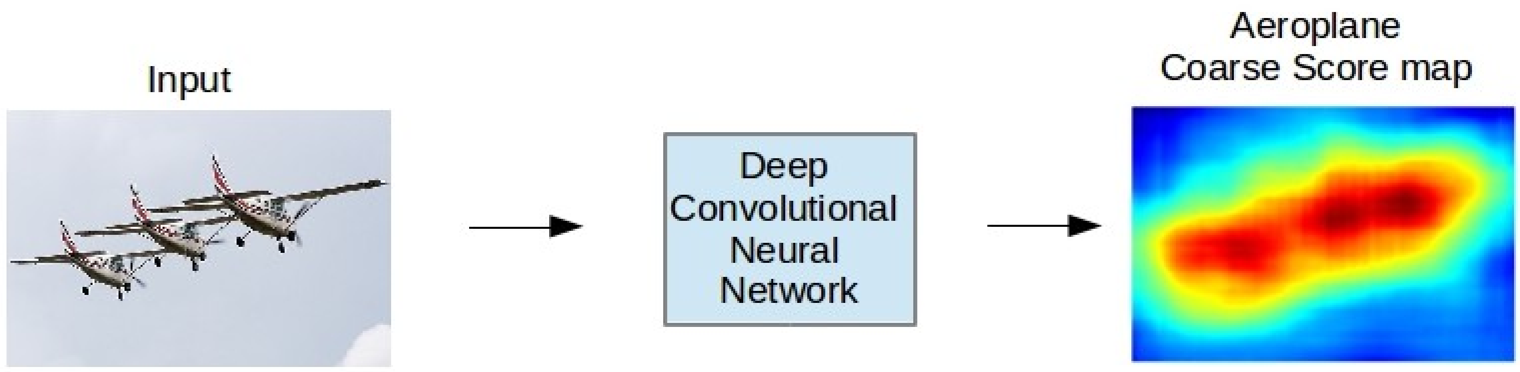


The U-net

- Add connections between layers of similar abstraction
- The idea is similar to gaussian/laplacian pyramids



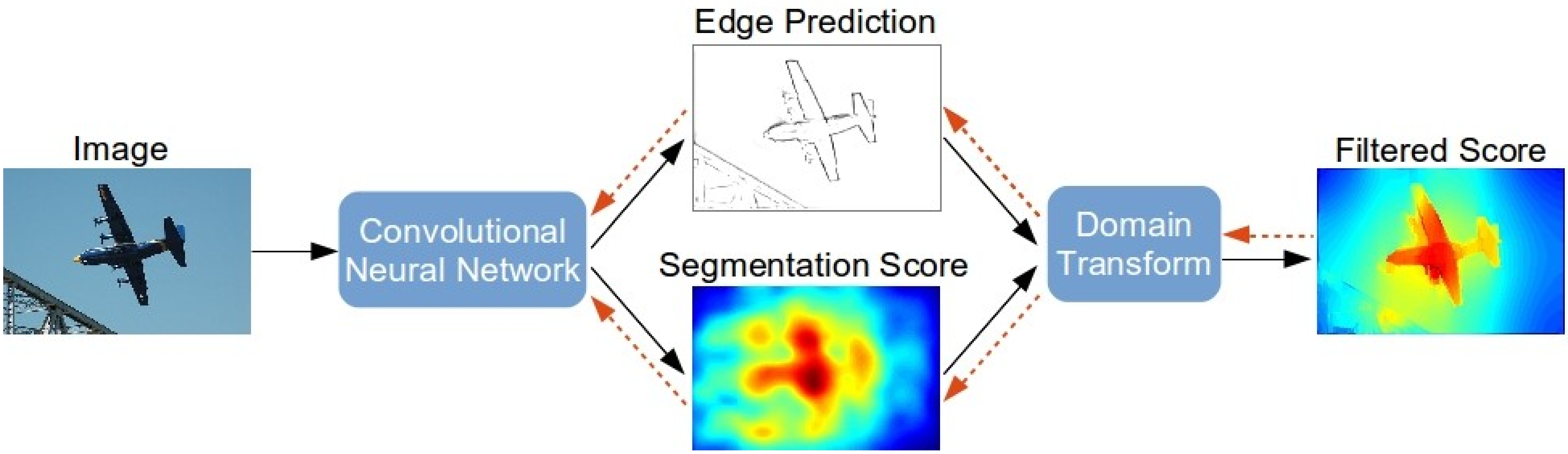
Raw network output is usually rather blurred



How can we get sharper edges?

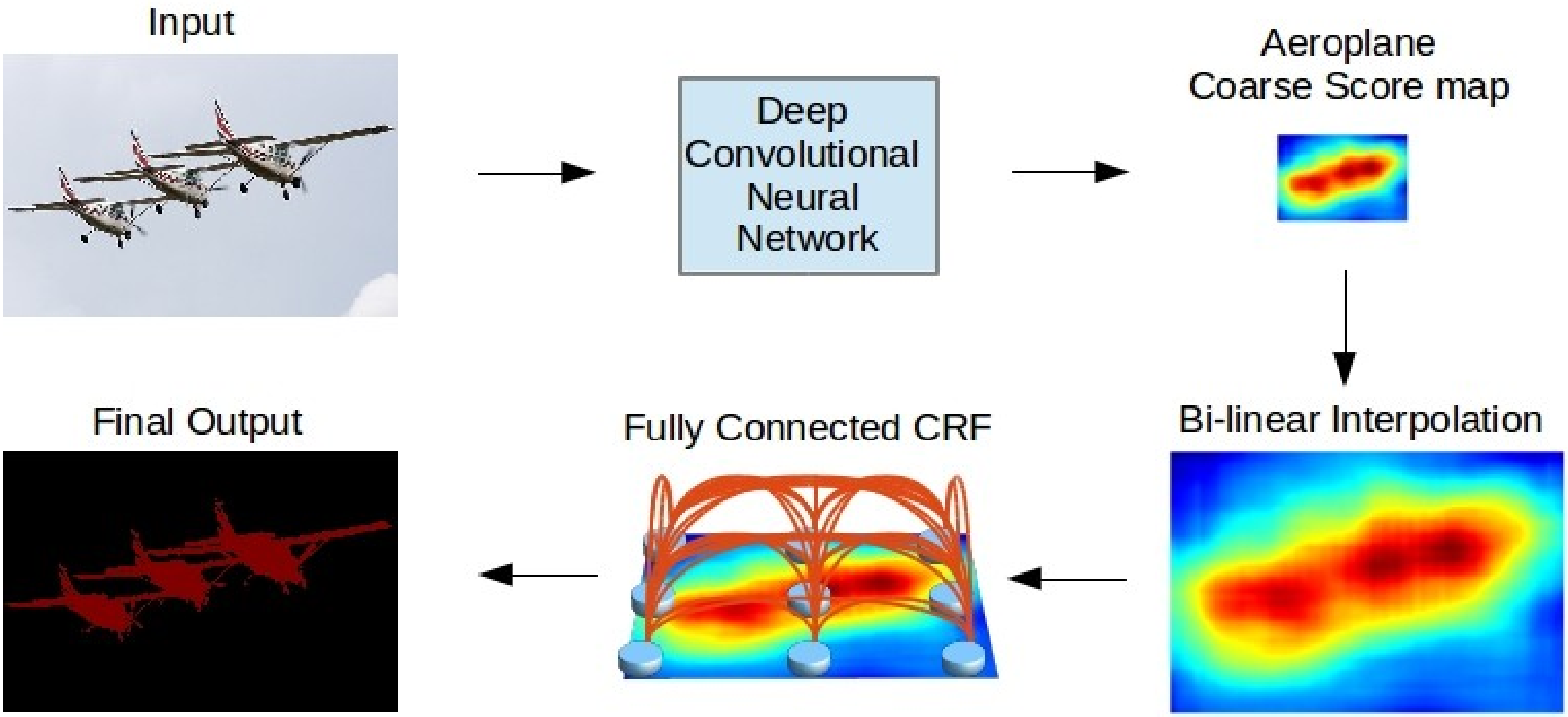
Segmentation: Edge prediction

- Train both class predictor and edge predictor



Segmentation: Graph models

- Tldr incentivize predicting same class over same texture / different classes over edges



See Vetrov's course or https://www.youtube.com/watch?v=uAsys22y5mY

More on that in your Graphical Models course

Image coloring

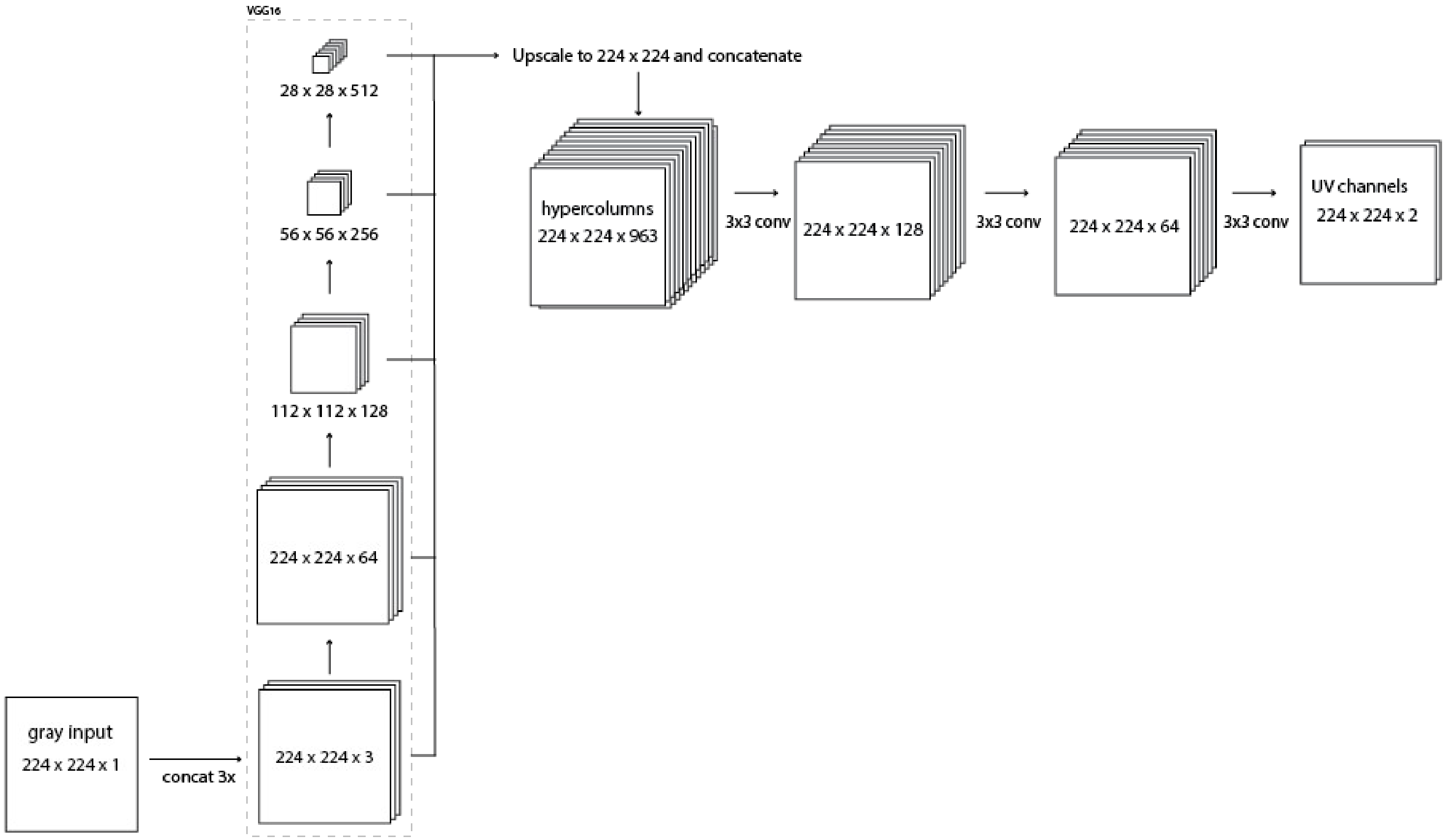
- Input:
  - gray/sepia image
  - mb neighboring frames
- Output:
  - RGB image
  - Same size as input
- **Ideas?**
  - Data?
  - Loss?
  - Architecture?

Validation X / prediction / y

http://tinyclouds.org/colorize

Image coloring: hypercolumns

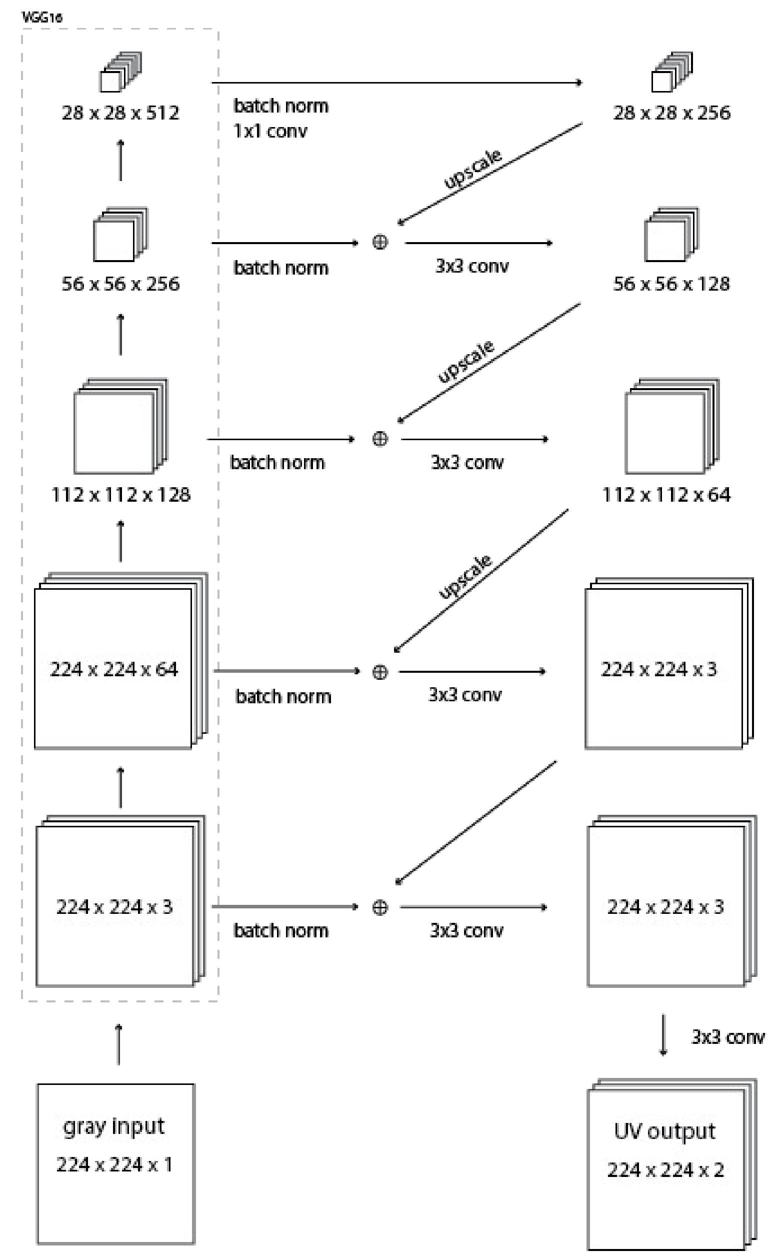
- Idea: upscale all layers to image size and pixel-wise
- Predict 2 UV channels
- Grayscale & UV→ RGB



http://tinyclouds.org/colorize

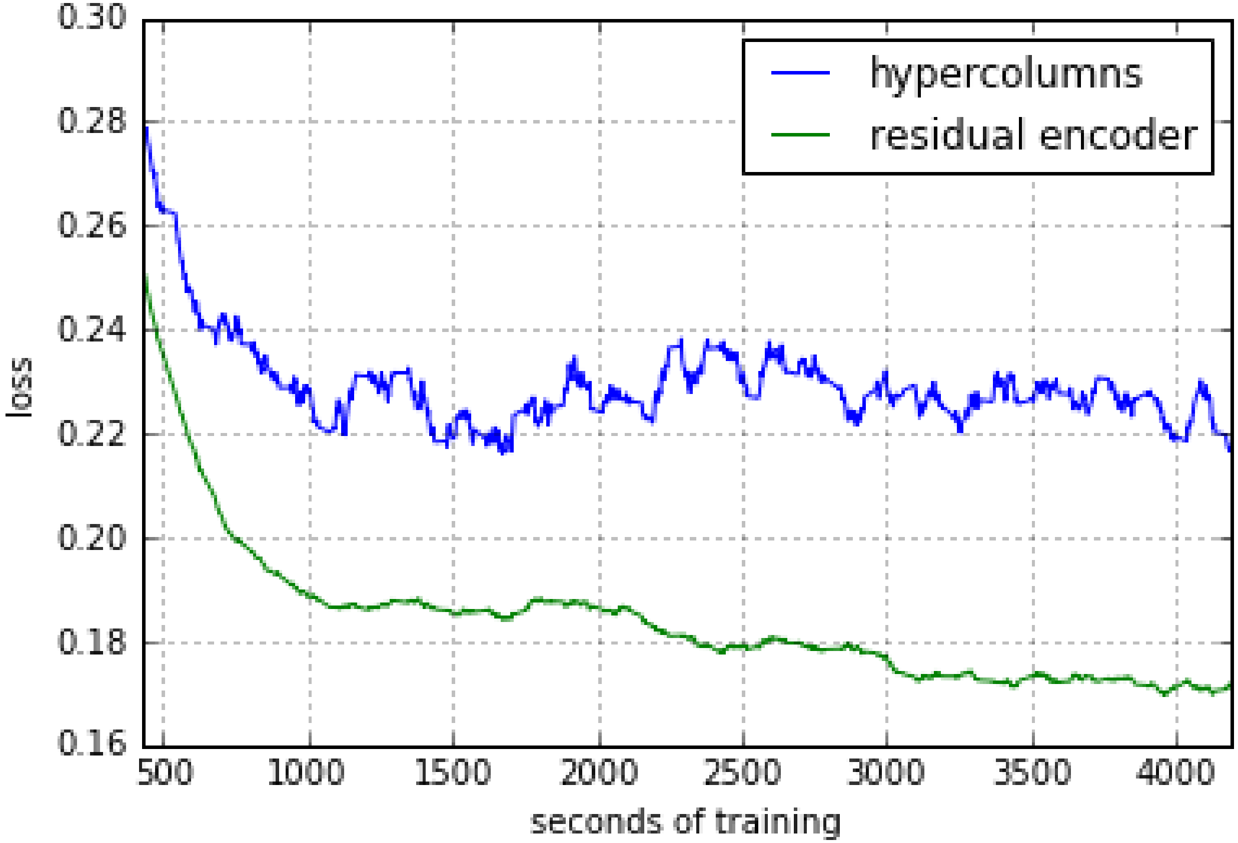
Image coloring: residual

- Similar to U-net
- Add instead of concat
  - Like ResNet
- Consumes less memory
  - No W x H x 1k blob



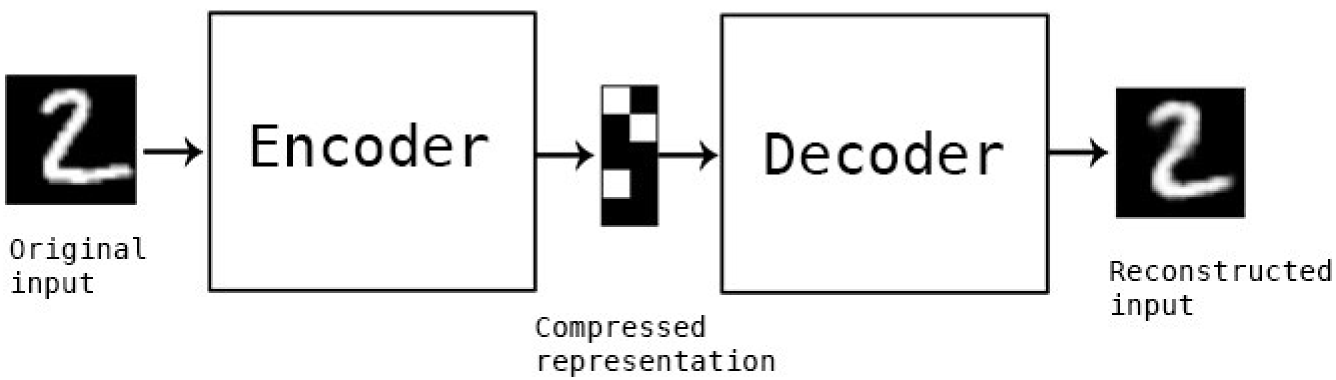
http://tinyclouds.org/colorize

Comparing these 2



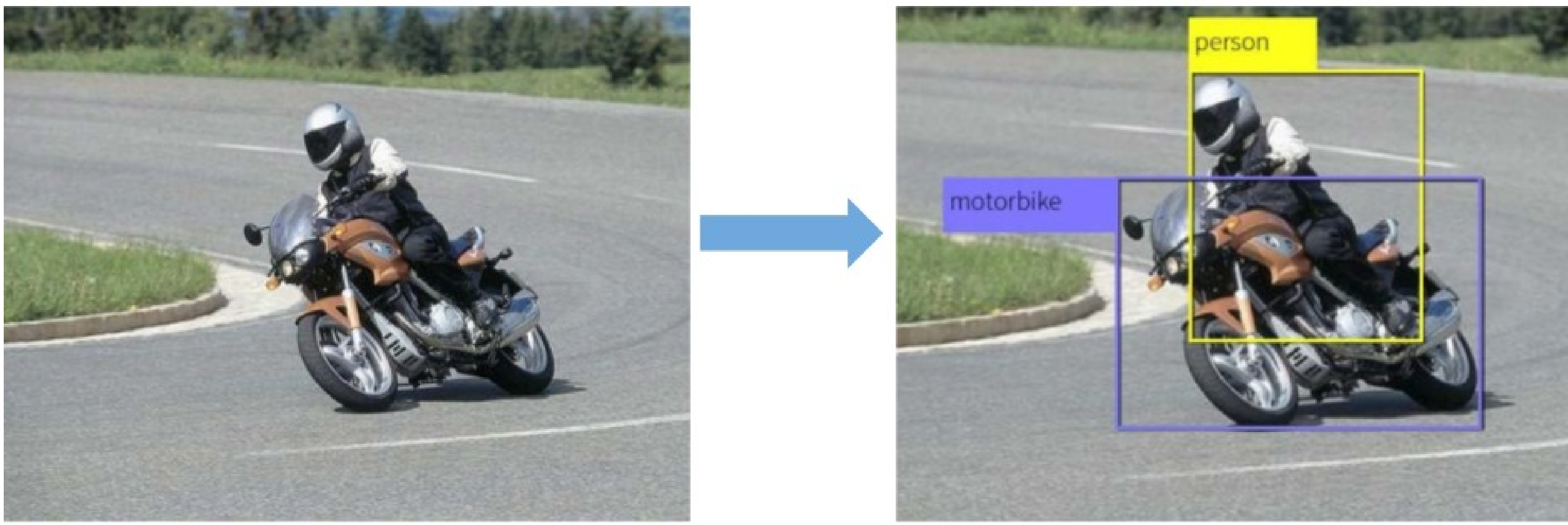
Autoencoders, briefly

- image → code → image
- Previous tricks with architecture still valid
- A few super cool task-specific hacks



More on that next week

Bounding Box Regression

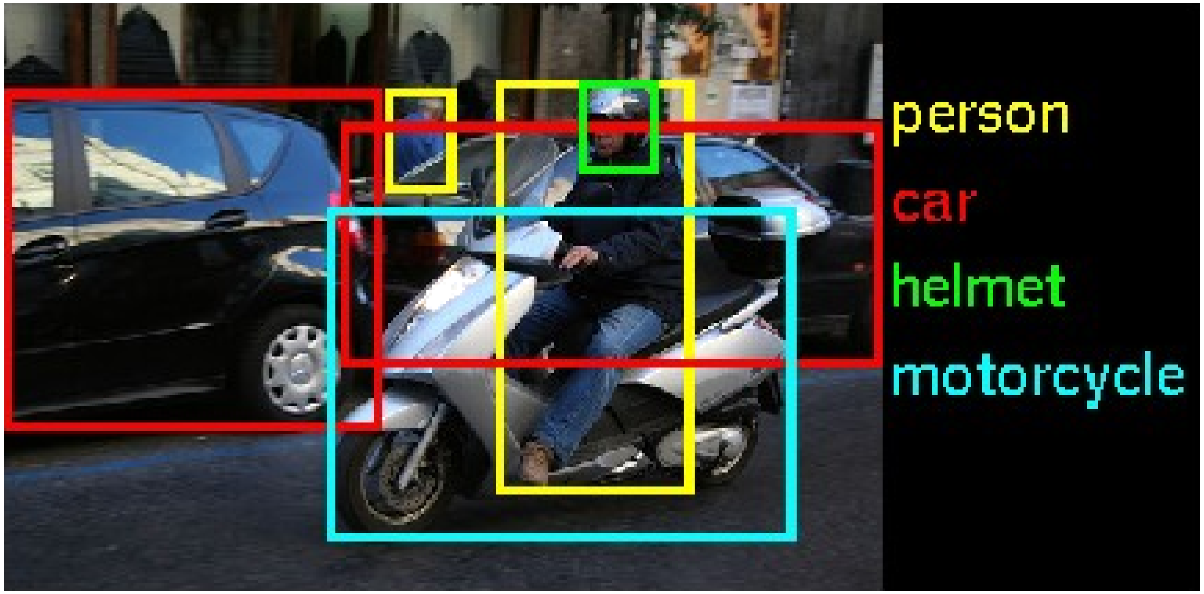


Input image　　　　Desired output

- Predict bounding rectangle for the object
- How do we deal with a single square object out of 100 possible objects?

Ideas?

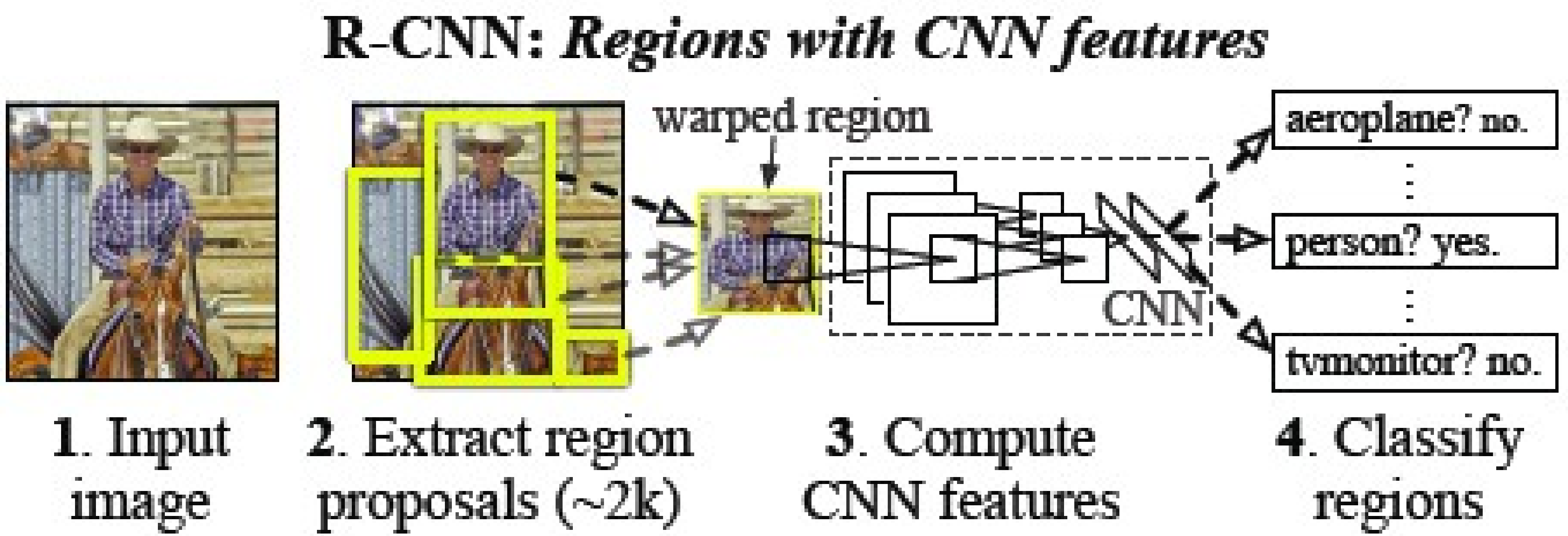
Bounding Box Regression



- Multiple squares?
- Non-square shape?
- Arbitrary rotation?

RCNN

- Generate candidates
- Adjust region shapes
- Apply classifier to each region



Bounding Box Adjustment

- Learn to predict how to adjust bounding box

  (dx, dy, dw, dh)

- Train on random candidates / RCNN output
- Iteratively adjust until good enough
- Learn the adjustment **policy**

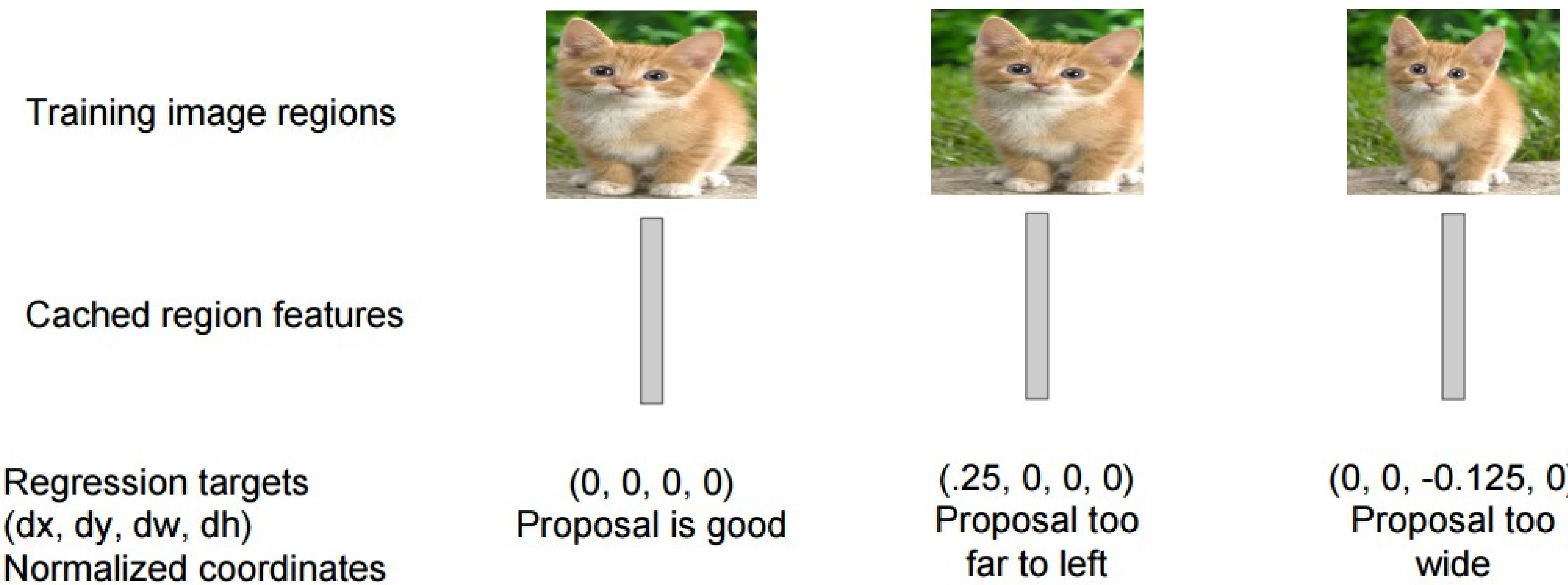
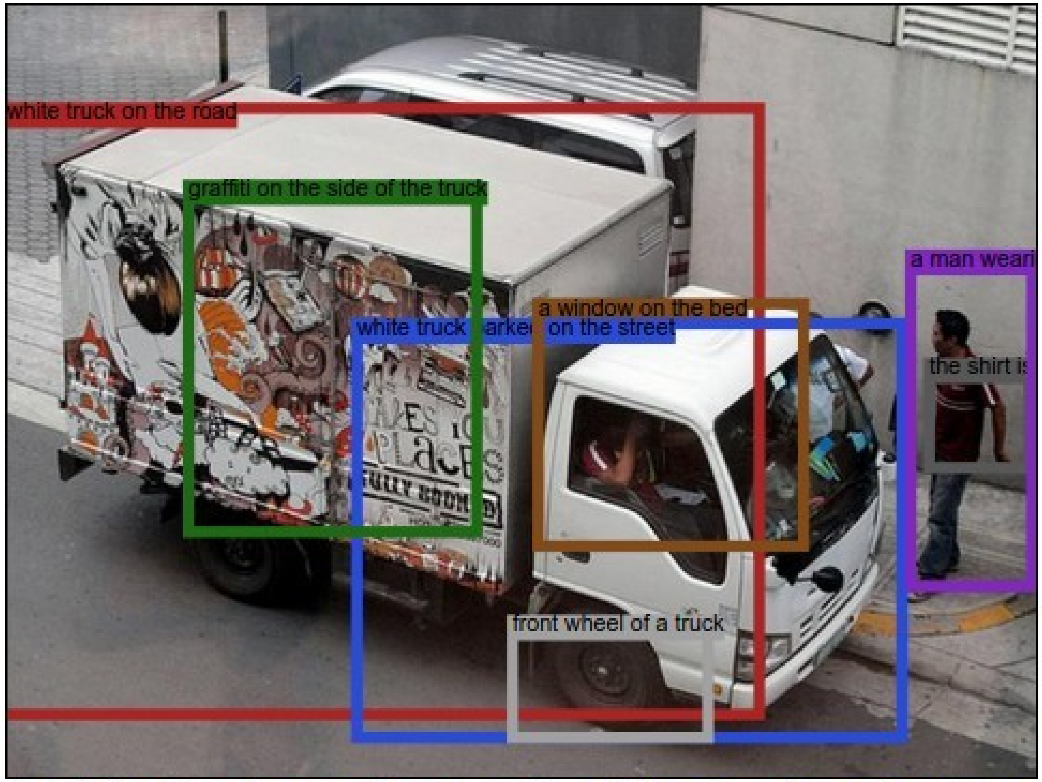


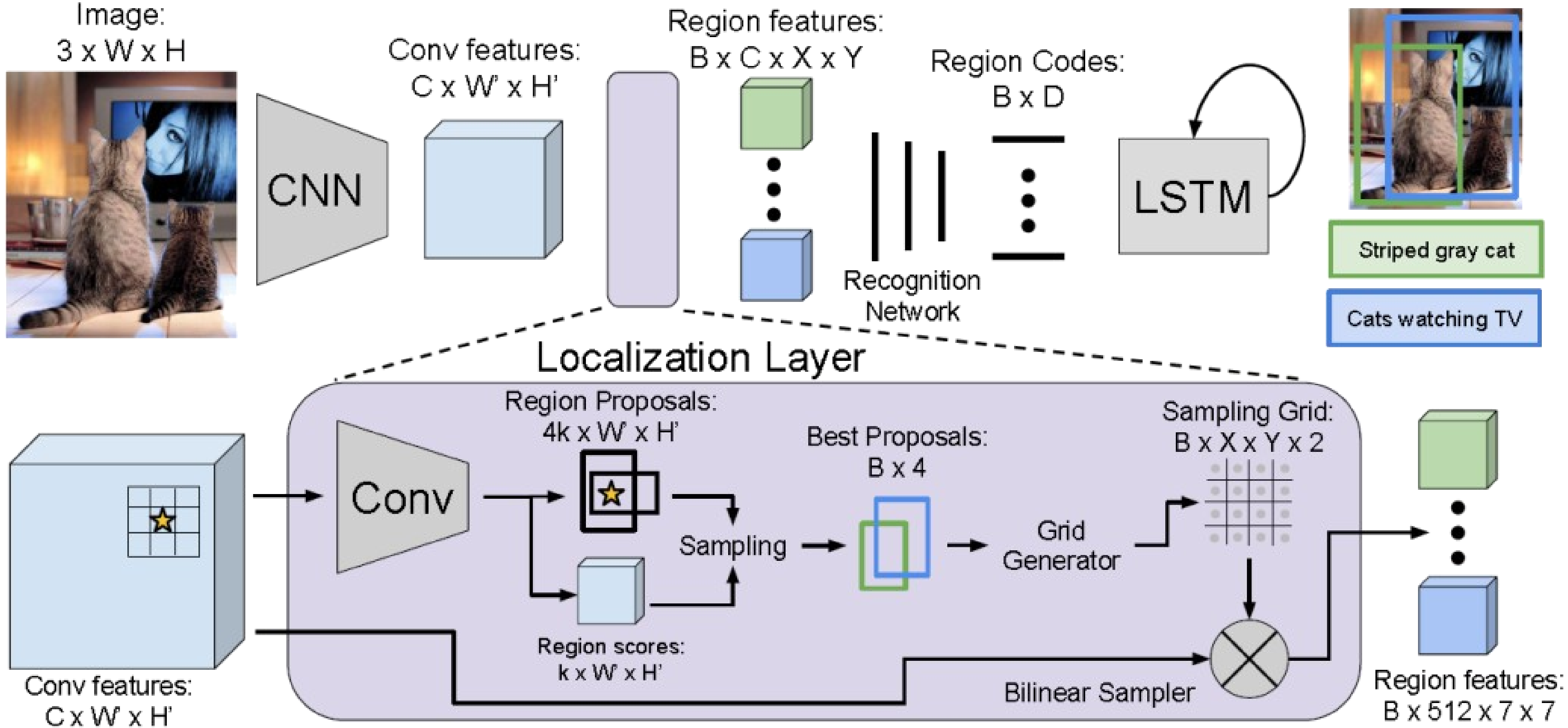
Image description / captioning



a white truck on the road. white truck parked on the street. the shirt is red. graffiti on the side of the truck. a window on the bed. a man wearing a black shirt. front wheel of a truck.

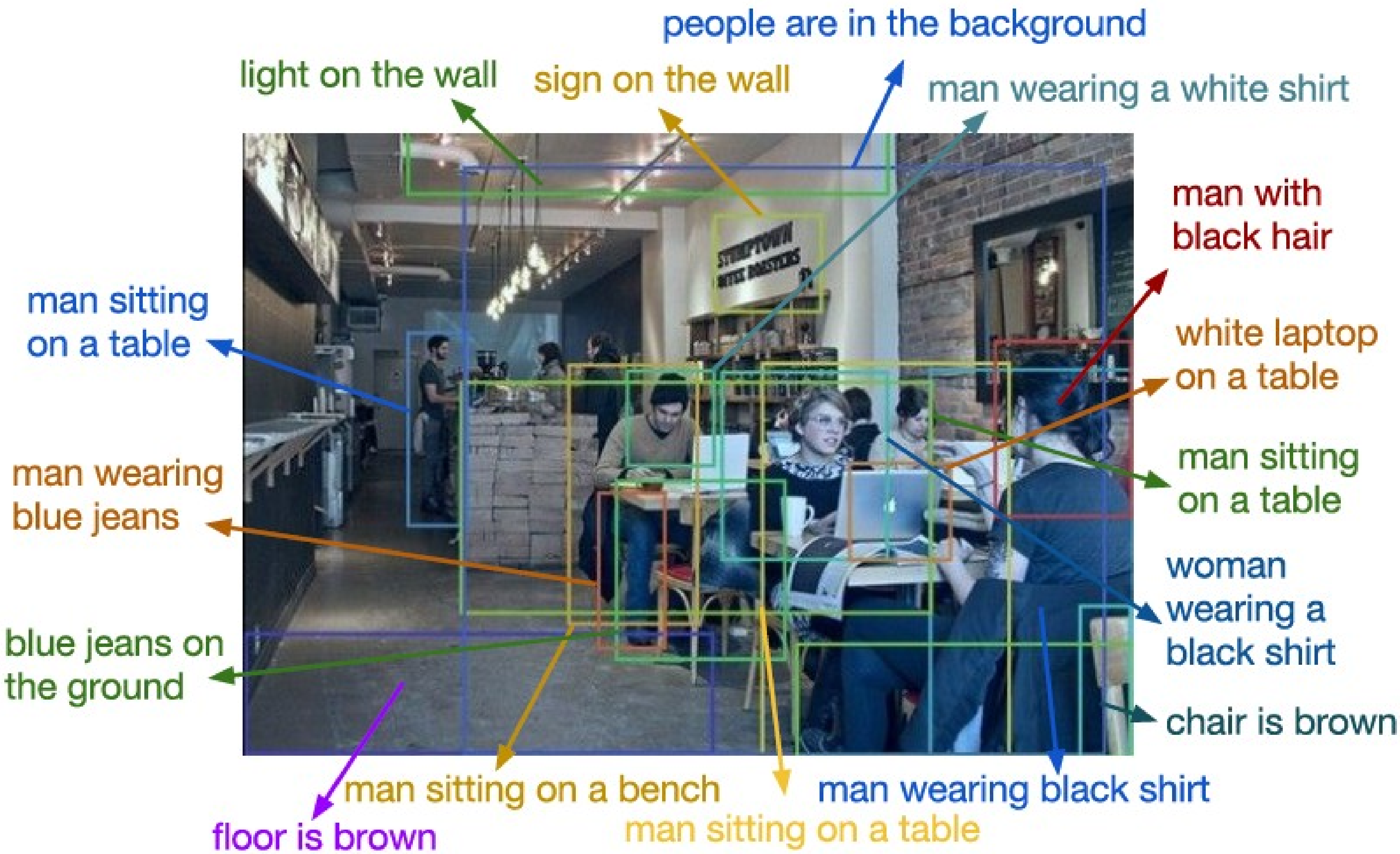
How could we approach this one?

Image description / captioning



Combines EVERYTHING we learned so far

Image description / captioning

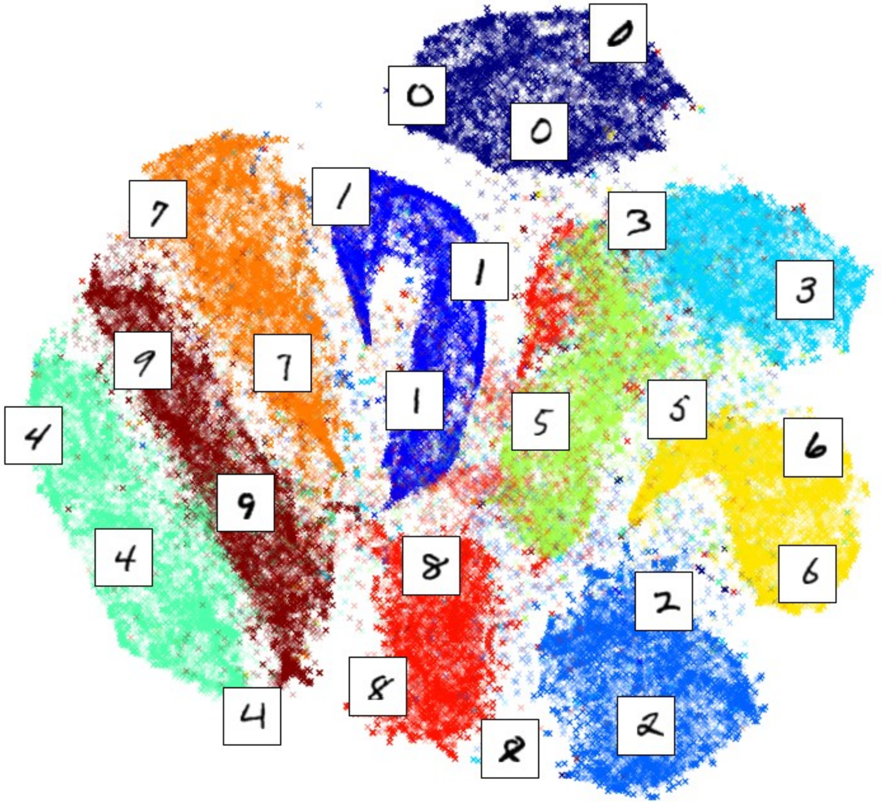


Representation learning

- Layer activations = representations
- Word2vec = word representation
- CNN activation = image2vec
- Image captioning = common space for images and text

MNIST dataset

Two-dimensional embedding of 70,000 handwritten digits with t-SNE

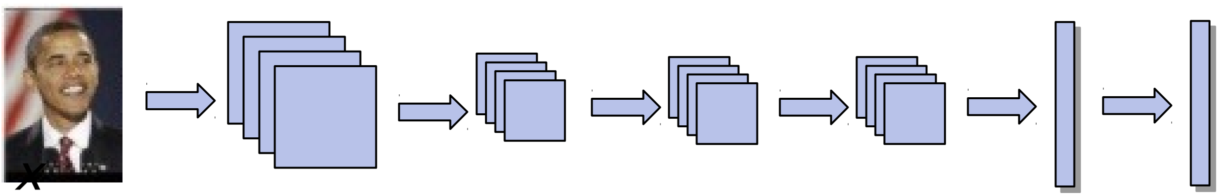


Metric learning

- Deliberately train representations that satisfy some properties
- **Word2vec:** words that appear in similar context should have similar vectors;
- **Image verification:** photos of the same person should have similar vectors, different persons = farther vectors;
- **Image captioning (alternative):** image must be close to it's correct descriptions, far from incorrect ones;
- One way: close ~ cosine, far ~ 1 - cosine

Image verification

- Image2vec



$$f_\theta(x) \in \mathbb{R}^n$$

- **We want:**

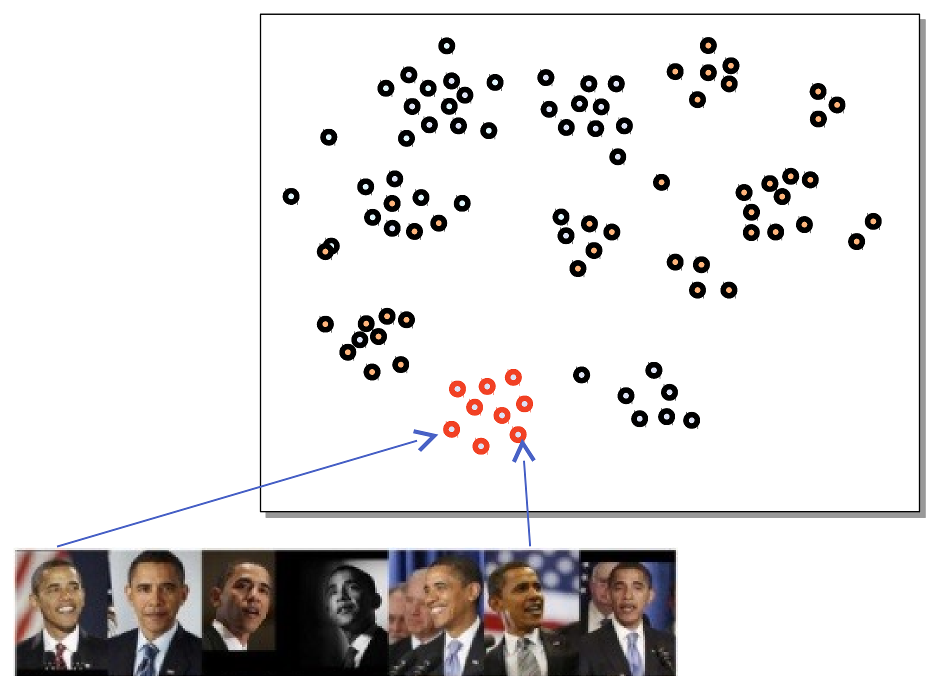
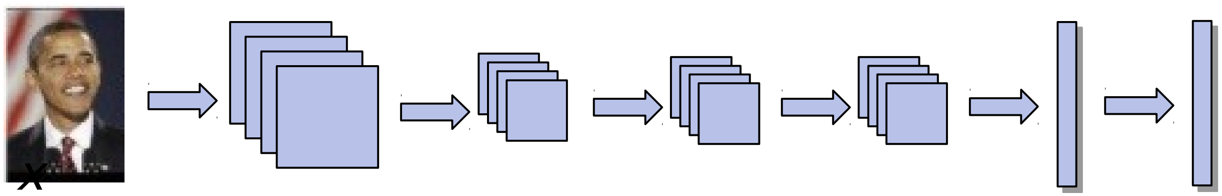


Image verification

- Image2vec



$$f_\theta(x) \in \mathbb{R}^n$$

- Similarity metric(example)

$$M(x1, x2) = \cos(f_\theta(x1), f_\theta(x2))$$

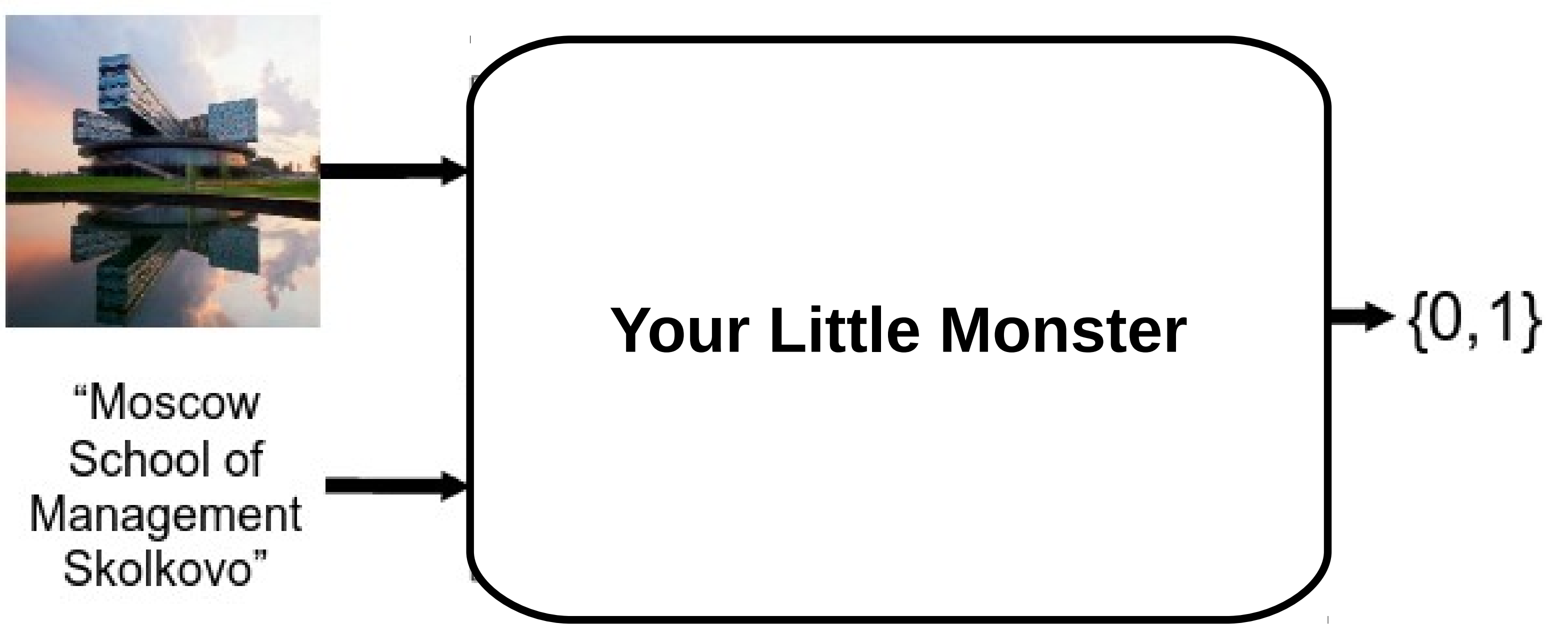
- x1, x2 - pair of images, T = #same person

Loss:

$$L = T \cdot M(x1, x2) + [1 - T] \cdot [1 - M(x1, x2)]$$

Case Study: Image search

- Naive approach: (image,query) → relevance
- Problem: need O(images x queries) runs
- Main objective = top 5~10 candidates



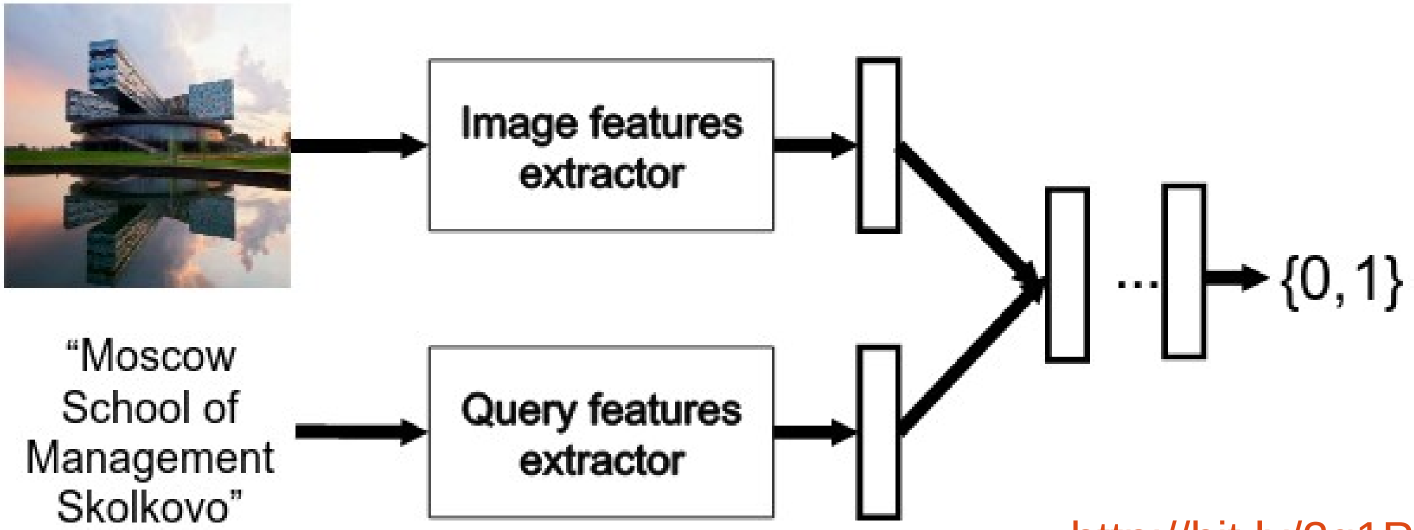
Case Study: Image search

Triplets (query, good image, bad image)

$$\left(q, i^{good}, i^{bad}\right)$$

Pairwise hinge loss

$$L = max\left(0, \delta - M\left(q, i^{good}\right) + M\left(q, i^{bad}\right)\right)$$



http://bit.ly/2g1PdWs

Hard negatives

- Baseline: pick negative images at random
  - Most cases are far too simple
- Idea: deliberately pick hardest negative images
  - Hardest = highest score with current NN

racehorse

Hard negatives

- Baseline: pick negative images at random
  - Most cases are far too simple
- Idea: deliberately pick hardest negative images
  - Hardest = highest score with current NN

Pick this one

http://bit.ly/2g1PdWs

Speed up:

- Precompute all query vectors O(n queries)
- Precompute all image vectors O(n images)
- Compute cosine in prediction time
- Locally Sensitive Hashing

...

Google
Yandex

Similar problems

- Other Information Retrieval problem
- Recommendation
- Banner ads
- Classification with a LOT of classes

Brace yourselves