Introduction to Deep Learning

LECTURE 4

Elad Hoffer
Nir Ailon
Ran El-Yaniv

Technion Israel Institute Of Technology,
Computer-Science department

Overview

1. Transfer learning using CNNs
2. Adapting CNNs to scale
3. Localization, Detection & Segmentation
4. Generative CNNs

Transfer learning using CNNs

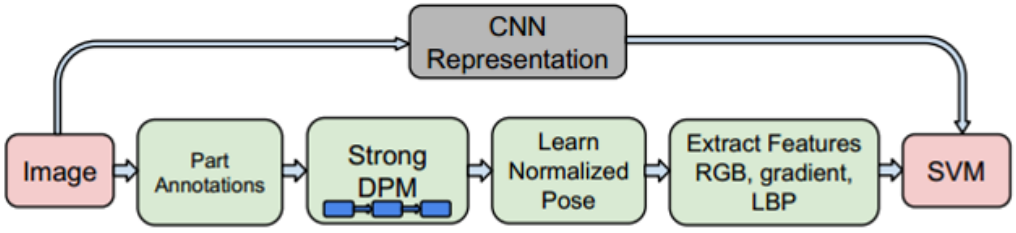
Transfer learning considerations using CNNs

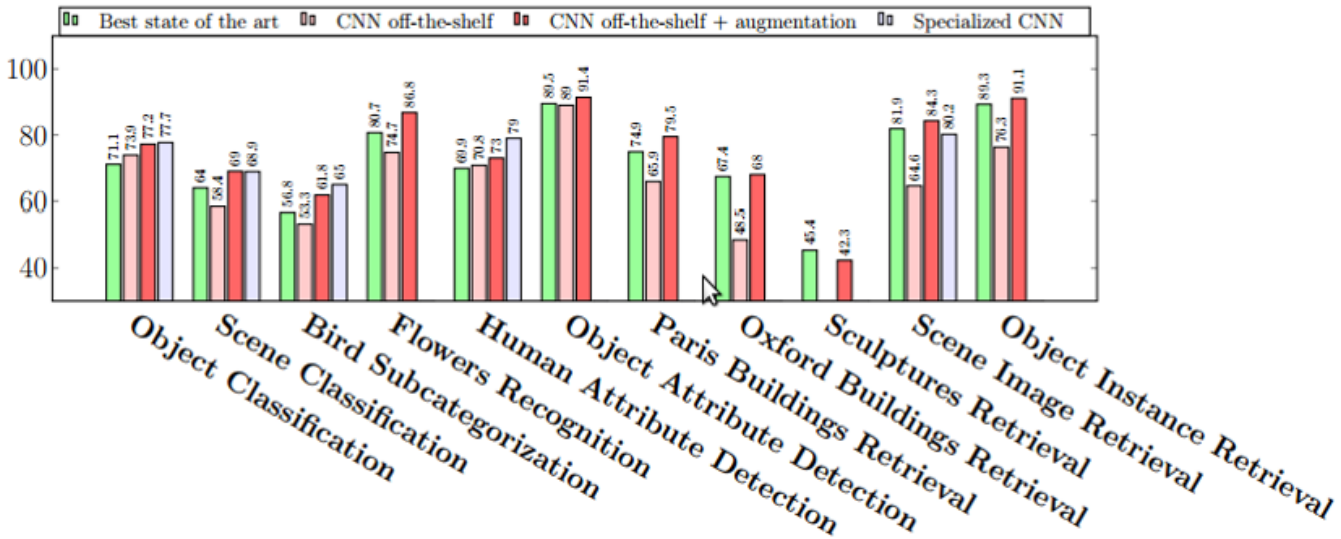
We've seen how, given enough data, we can train a convolutional network to learn complicated visual tasks.

- In practice, very few people train an entire Convolutional Network from scratch (with random initialization), because it is relatively rare to have a dataset of sufficient size.
- Instead, it is common to use a network that was pretrained on a very large dataset (e.g. ImageNet, which contains 1.2 million images with 1000 categories),

Transfer learning

This turned out to work pretty well in practice:





Transfer learning considerations using CNNs

When transfering our model to the new task, our two main approaches for using the trained CNN are

- *Generic feature extractor* - Take the pretrained network, remove the last fully-connected layer, then treat the rest of the ConvNet as a fixed feature extractor for the new dataset. You can then train a linear classifier (e.g. Linear SVM or Softmax classifier) on those features for the new dataset.
- *Fine-tuning* - Replace the classifier with one for the new task, then fine-tune the weights of the pretrained network by continuing the backpropagation. It is possible to fine-tune all the layers of the ConvNet, or it's possible to keep some of the layers fixed.

Transfer learning considerations

The extent of how many layers are taken from the pretrained network, and whether to fine-tune them is dependent on various aspect

- The relation between original and new data - how much do the tasks differ
- The amount of new data to train on - should we be worried about overfitting our new training data?

This will also affect the training regime - learning rate for fine-tuned layers

Transfer learning considerations

- *Generality vs specificity* - earlier features of a ConvNet contain more generic features (e.g. edge detectors or color blob detectors) that should be useful to many tasks, but later layers of the ConvNet becomes progressively more specific to the details of the classes contained in the original dataset.
- *Catastrophic forgetting* - networks can behave very bad when the training data distribution changes abruptly. For example, using only a subset of classes, or changing the learned task.

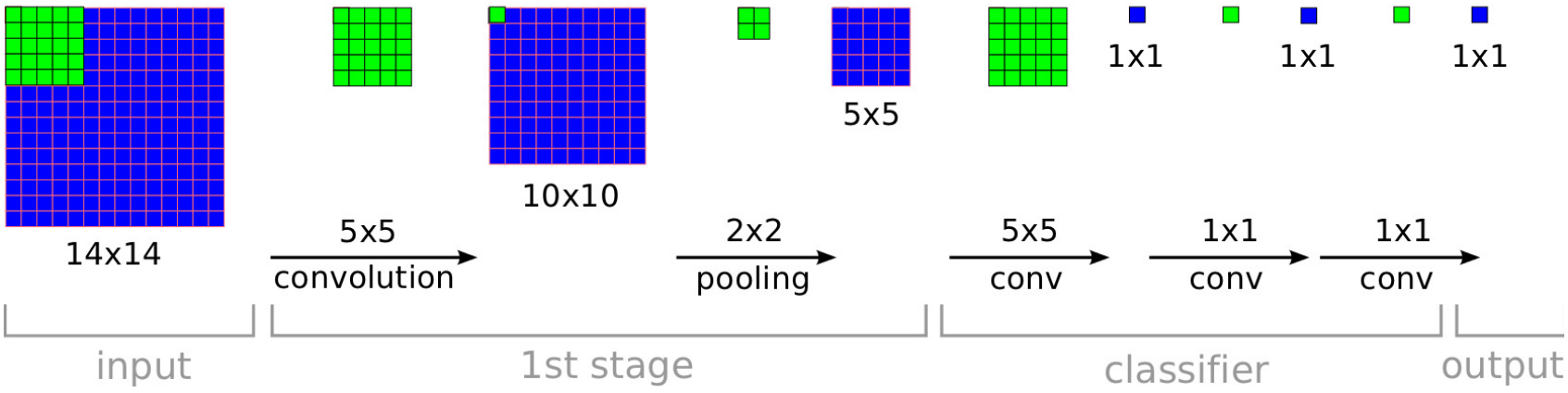
Adapting CNNs to scale

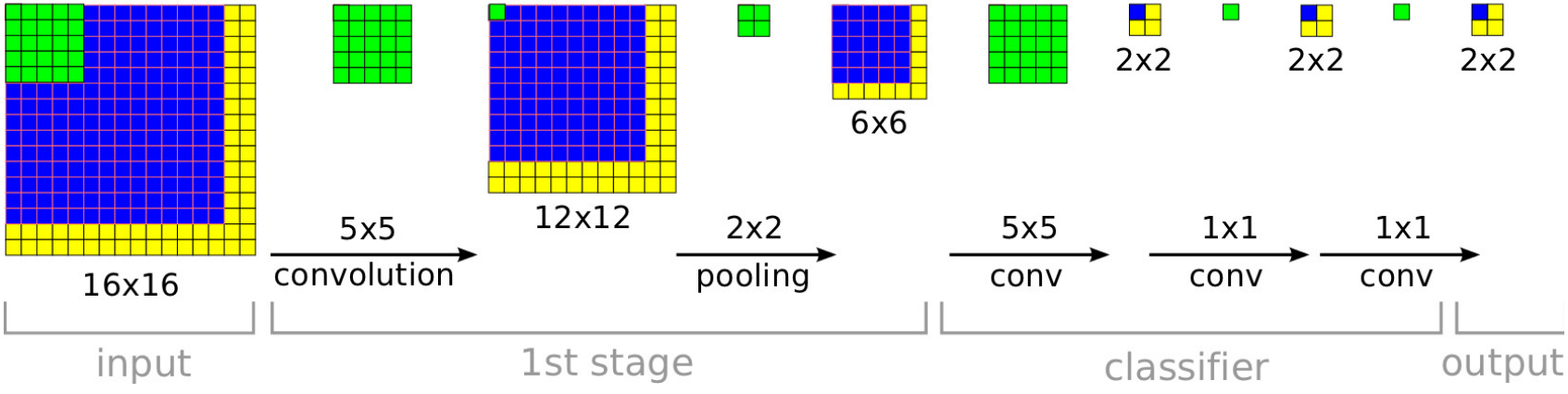
Spatial classification

In order to classify images bigger than the size we trained on, we can view fully-connected layers as convolutional kernels:

- Replace the first fully-connected layer that looks at spatial region (e.g 7×7) with a convolutional layer with kernel of the same size.
- Replace each subsequent layer with a convolutional layer with kernel of 1×1.
- Instead of a single classification, we will receive a class-map, where each spatial location is related to a different region in input image

Spatial classification

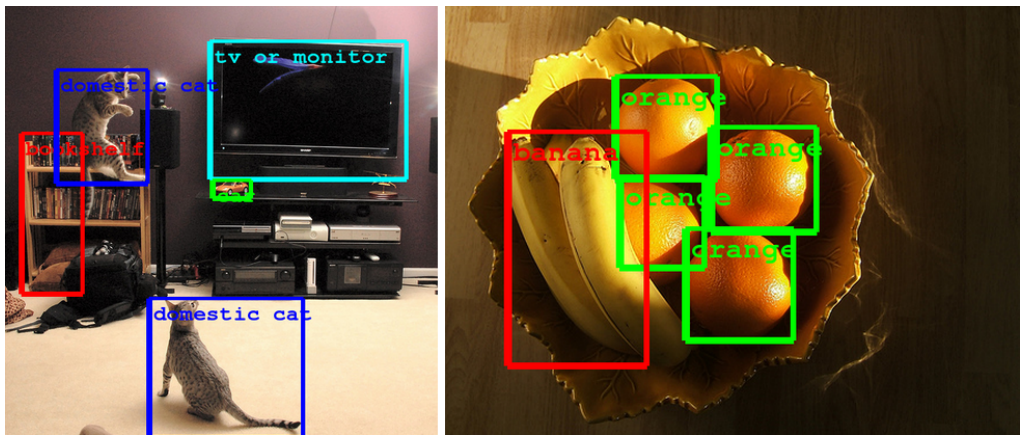




Localization, Detection & Segmentation

Localization + Detection

As we've seen, we can adapt a trained network for new sizes ands scales. A natural additional tasks in computer-vision is to try to *localize* and *detect* objects in an image.



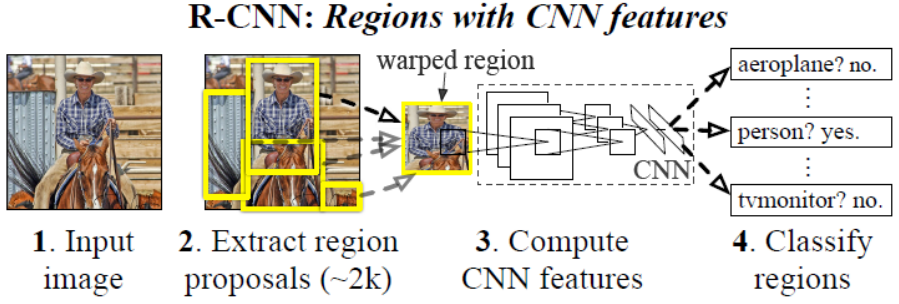
We need both to classify our object and to localize it

- For detection - we also need to handle multiple objects and reject a lot of false-positives (not-an-object)

Localization + Detection with CNNs

Some methods for detection using CNNs when we posses ground-truth bounding boxes

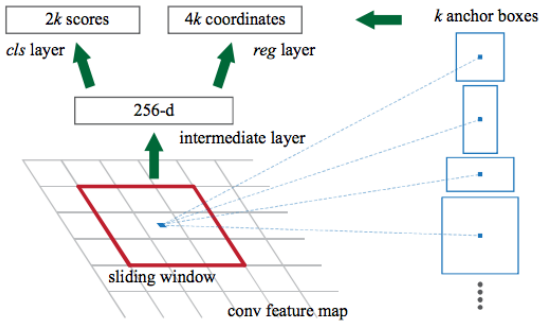
- Most naive approach is to add a regression objective - estimating the bounding boxes of each object (OverFeat - Sermanent 13').
- A more popular and effective method is to use *region-proposals* (often using "selective-search"), suggested in RCNN (Girshick 14')

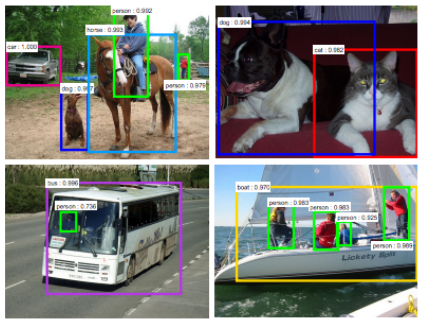


Localization + Detection with CNNs

The RCNN approach, although accurate, had a bottleneck - generating and evaluating on ~ 2000 proposals. To remedy this, future works suggested

- Evaluating most of the network on entire image, and pooling the proposals from feature space instead of image space (SPP, Fast-RCNN)
- Moved the region-proposal to be an inherent part of the network (Faster-RCNN)

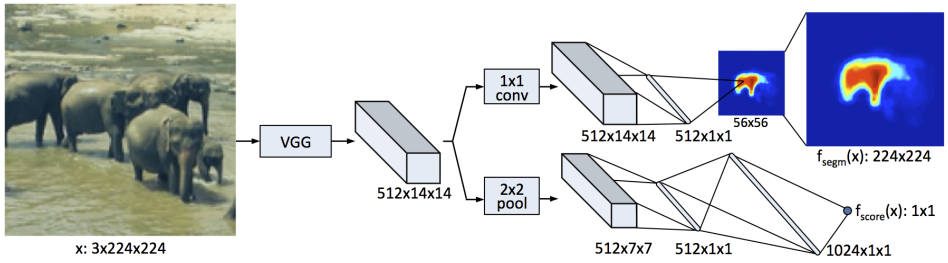




Segmentation using CNNs

Another important computer-vision task is to segment whole images. This can be viewed as a classification at the pixel level.

- We might wish to incorporate both global, abstract features, as well as localized low-level features
- This means using multiple layers for final segmentation (Long 14')
- As with detection, trend is to build whole end-to-end models by creating object-candidates (Pinheiro 15')



Generative CNNs

Generative adversarial networks

We mostly talked about supervised, discriminative models (with the exception of auto-encoders).
One new and and exciting framework for unsupervised, generative models is the *Adversrial network* (Goodfellow 14').
It consists of two networks trained simultaneously:

- A *generative* model G that tries to capture the data distribution and generate new samples.
- A *discriminative* model D that estimates the probability that a sample came from the training data rather than G.

The training procedure for G is to maximize the probability of D making a mistake - it is "cheating" by using the gradients with respect to D.

Generative adversarial networks

"Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks" (Alec Radford, Luke Metz, Soumith Chintala 15')

Generative adversarial networks