Rich feature hierarchies for accurate object detection and semantic segmentation

Ross Girshick *et al.*

2014

1 Contributions

- Solve the CNN localization problem by operating within the "recognition using regions" paradigm.
- Show that *supervised* pre-training on a large auxiliary dataset (ILSVRC), followed by domain-specfic fine-tuning on a small dataset (PASCAL), is an effective paradigm for learning high-capacity CNNs when data is scarce.

2 Object detection with R-CNN

2.1 Module design

Our object detection system consists of three modules.

1. The first generates category-independent region proposals. These proposals define the set of candidate detections available to our detector.
2. The second module is a large convolutional neural network that extracts a fixed-length feature vector from each region.
3. The third module is a set of class-specific linear SVMs.

2.1.1 Region proposals

We use **selective search** here.

- Extract around 2000 region proposals.
- Use "fast mode" in all experiments.

2.1.2 Feature extraction

The CNN is the *Caffe* implementation based on **AlexNet**. Features are computed by forward propagating a mean-subtracted 227×227 RGB image through five convolutional layers and two fully connected layers.

Steps of pre-processing region proposals:

- Dilate the tight bounding box so that at the warped size there are exactly p pixels of warped image context around the original box (we use $p = 16$).
- Warp all pixels in a tight bounding box around it to the required size (227×227).

2.1.3 Classification

For each class, we score each extracted feature vector using the SVM trained for that class. Given all scored regions in an image, we apply a **greedy non-maximum suppression** (for each class independently) that *rejects* a region if it has an *intersection-over-union* (IoU) overlap with a higher scoring selected region larger than a learned threshold.

2.2 Run-time analysis

Two properties make detection efficient:

1. All CNN parameters are shared across all categories.
2. The feature vectors computed by the CNN are low-dimensional (4k).

The result of such sharing is that the time spent computing region proposals and features (13s/image on a GPU or 53s/image on a CPU) is amortized over all classes.

2.3 Training

2.3.1 Supervised pre-training

We discriminatively pre-trained the CNN on a large auxiliary dataset (ILSVRC2012 classification) using *image-level annotations* only (bounding-box labels are not available for this data).

2.3.2 Domain-specific fine-tuning

We continue training of the CNN parameters using only warped region proposals:

1. Replace the original classfication layer with a randomly initialized classification layer.
2. Use smaller learning rate (0.001) to avoid clobbering the initialization.

2.3.3 Object category classifiers

Carefully choose IoU overlap threshold (0.3, selected by *grid search*), below which regions are defined as negatives (others positives).

Once features are extracted and training labels are applied, we optimize one linear SVM per class. Since the training data is too large to fit in memory, we adopt the standard hard negative mining method.

2.4 Results

- 53.7 mAP on VOC 2010 test.
- 53.3 mAP on VOC 2011/12 test.
- 31.4 mAP on the ILSVRC 2013 competition.

3 Visualization, ablation, and modes of error

3.1 Visualizing learned features

We propose a simple (and complementary) non-parametric method that directly shows what the network learned.

1. Single out a particular unit (feature) in the network and use it as if it were an object detector in its own right.
2. Compute the unit's activations on a large set of held-out region proposals (about 10 million).
3. Sort the proposals from highest to lowest activation.
4. Perform non-maximum suppression.
5. Display the top-scoring regions.

The network appears to learn a representation that combines a small number of class-tuned features together with a distributed representation of shape, texture, color, and material properties.

3.2 Ablation studies

3.2.1 Performance layer-by-layer, without fine-tuning

We start by looking at results from the CNN *without fine-tuning* on PASCAL, i.e. all CNN parameters were pre-trained on ILSVRC 2012 only.

- Features from fc_7 generalize worse than features from fc_6.
- Removing *both* fc_7 and fc_6 produces quite good results.

Much of the CNN's representational power comes from its convolutional layers, rather than from the much larger densely connected layers.

3.2.2 Performance layer-by-layer, with fine-tuning

We now look at results from our CNN after having fine-tuned its parameters on VOC 2007 trainval.

The boost from fine-tuning (increases mAP by 8.0 percentage points to 54.2%) is much larger for fc_6 and fc_7 than for pool_5 ,which suggests that the pool_5 features learned from ImageNet are general and that most of the improvement is gained from learning domain-specific non-linear classifiers on top of them.

3.2.3 Comparison to recent feature learning methods

All R-CNN variants strongly outperform the three DPM baselines (DPM V5, DPM ST, DPM HSC), including the two that use feature learning.

3.3 Network architectures

We have found that the choice of architecture has a large effect on R-CNN detection performance. R-CNN with **VGG16** substantially outperforms R-CNN with **AlexNet**, increasing mAP from 58.5% to 66.0%.

However, the forward pass of VGG16 taking roughly 7 times longer than AlexNet.

3.4 Detection error analysis and bounding-box regression

We applied the excellent detection analysis tool from Hoiem et al. in order to reveal our method's error modes, understand how fine-tuning changes them, and to see how our error types compare with DPM.

Based on the error analysis, we implemented a simple method inspired by DPM to reduce localization errors.