Timely R-CNNs

Sergey Karayev
UC Berkeley

Ross Girshick
UC Berkeley

Sergio Guadarrama
UC Berkeley

Mario Fritz
MPI Informatics

Trevor Darrell
UC Berkeley

Abstract

Recently, an object detection architecture based on two key decisions has demonstrated an undeniable increase in performance on both PASCAL VOC and ILSVRC detection datasets. The first is doing bottom-up Region of Interest (ROI) proposals instead of sliding-window evaluation. The second is using multi-layer Convolutional Neural Nets (CNNs) pre-trained on ILSVRC classification data to featurize canonically resized ROIs. Although by far the best in terms of recognition performance, the method is computationally costly, requiring processing of several hundred of thousand regions with the CNN. Even with efficient GPU implementation, the method takes ˜10 s per image after the computation of region proposals (another ˜2 s).

We propose three novel speed-ups for the task. (1) Dense evaluation: the whole image is processed with a CNN up to the highest non-fully-connected layer, and ROIs are featurized by cropping and resizing that highest layer, for use as an initial filter. (2) Cascaded CNN: each ROI passing the initial filter is further processed by a CNN that is augmented with a *reject* option after each layer. (3) The selection of ROIs for further processing is done dynamically, taking into account the evaluation results on the ROIs selected so far.

Combined, these speed-ups allow us to match original R-CNN performance in 20% of the time.

1 Introduction

Multi-layered convolutional models have recently been shown to offer impressive levels of performance on visual detection tasks, by applying deep convnet models trained on object recognition tasks to multiple candidate image windows in a scene [1, 2]. While convolutional models are inherently efficient to implement, as convolution itself is a parallelizable and long studied computational primitive, the amount of work to perform complete inference at all candidate region windows is considerable.

Further, unless care is taken in the design of the computation, much of the computation can be repetitive. Overfeat overcomes this by incrementally computing the convnet features in overlapping windows, but then restricts the window locations to be on a relatively regular grid. RCNN allows more general region proposals guided by low-level segmentation cues ([3]), but repeats the computation of convolutional features across numerous overlapping windows. While RCNN has leading performance on PASCAL and ImageNet Detection as of the date of this writing, it is rather slow, at XXs per image.

Existing detection schemes, including RCNN and Overfeat, are generally relatively naive in terms of the temporal properties of their inference. They take all regions as equal when performing search, and in contrast to human performance, do not consider any attentive or time-sequential aspects. In

this paper we advocate for such a perspective, and formulate "Timely" detectors which explicitly optimize over the order of regions to consider to maximize their efficacy over time. (Or when time is considered a costly resource). We build on the work of [4], who considered time-sensitive feature selection in a classification paradigm, but extend the model to be relevant to contemporary detector settings, and to include spatial reasoning.

Our model yields novel forms of "cascaded"-style processing for such detectors, where large amounts of computation is pruned when it is deemed redundant, or likely unproductive. We consider both static and dynamic policies for such behavior, and features based on low-level cues and mid-level convnet features. We focus on speeding up the RCNN framework, but our work is relevant to any detector that operates a relatively expensive function over a set of candidate locations. Our model implicily learns to look elsewhere when a nearby region is already examined, and to look above a motorbike for a possible person; it generalized well known principles of non-maximal suppression and inter-task context.

We consider three specific mechanisms, which are independently and jointly valuable. [[1) dense eval.; 2) cascade; 3) dynamic]]. Together we call these "Timely" methods for object detection, hence the title of our paper.

We discuss related work in section 2, cover all parts of our proposed method in section 3, and present evaluation results on the PASCAL VOC in section 4.

2 Related Work

Object recognition with CNN

- Alexnet [5]
- RCNN [2] and the couple other CNN detection methods [6] [7].
- Overfeat: explain dense evaluation [1]

Cascaded detection

- Viola Jones [8]
- Cascaded DPM [9]
- **todo**: Is there any CNN work?

Dynamic selection

- Timely Object Recognition [4]
- Cross-talk cascades [10]
- **todo**: What else?

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

3 Method

Our method builds on the architecture described in [2]. As summarized in Figure 1, R-CNN starts with external region-of-interest proposals (ROIs), and then does the following: (1) for each ROI,

classifies it with a CNN, obtaining multi-class scores; (2) post-processes the scored ROIs with non-maximal suppression and other bounding box refinement to obtain the final detections.

We aim to incorporate several speed-ups into a similar architecture. As summarized in Figure 2, we start with the same ROI proposals, and then (1) score each proposal with a quick-to-compute feature; (2) select the ROI that is most promising, and score it with a CNN, which is additionally sped up with a cascaded structure; (3a) either select another ROI, incorporating the scores of the first one into the selection – or (3b) post-process the ROIs scored so far and exit.

For the quick feature, we consider the CNN pixel gradient in subsection 3.1 and an amortized R-CNN variant that operates on a feature pyramid in subsection 3.2. The Cascaded CNN is described in subsection 3.3. The dynamic region selection is explained in subsection 3.4.

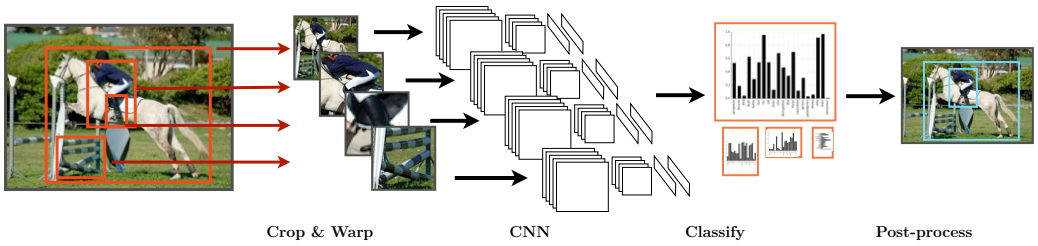

Figure 1: R-CNN architecture: image regions are cropped, resized, and each one fed through a CNN with classification layers. The classifier outputs are post-processed to give the final detections.

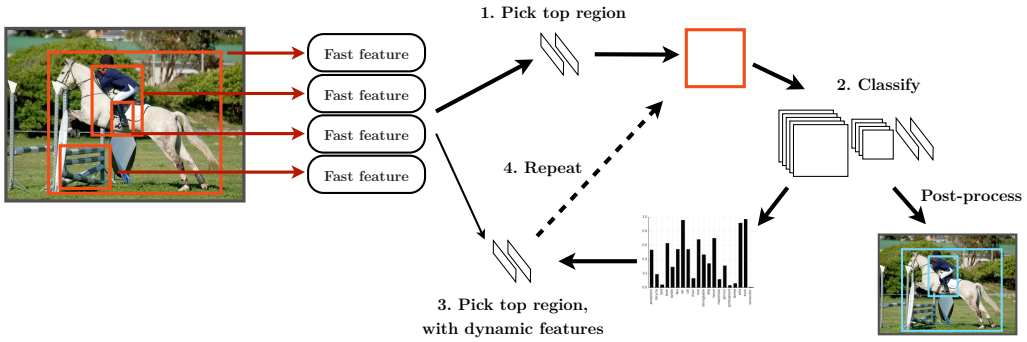

Figure 2: Dynamic region selection combines both speedups and introduces another.

3.1 CNN pixel gradient computation

One fast-to-compute feature we consider for the preliminary featurization of ROIs is the pixel gradient, back-propagated throgh the classifier CNN applied to the whole image. This gradient corresponds to a kind of saliency map onto the image [7], and can therefore be used to evaluate ROIs.

- **todo**: Explain backprop to pixels.
- **todo**: Explain how we featurize it: warp each ROI to canonical size and also include overall statistics, then train to classify > 0.3 overlap with ground truth.
- **todo**: FIGURE: shows schematic structure of the flow and gradient saliency maps for a couple of images.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

3.2 Pyramid amortized R-CNN

Another fast ROI featurization we consider is a variant of R-CNN that processes the whole image, at multiple scales, with the CNN up to the topmost non-fully-connected layer. We call this multiscale CNN output a "feature pyramid". As Figure 3 shows, ROIs are then cropped from the feature pyramid at the best matching scale, warped to a canonical size, and classified. ROIs are highly overlapping, but their featurization is shared through the pyramid, amortizing its construction cost. This doesn't work as well as the original R-CNN system, but is much faster, and can therefore be used as the fast feature for the dynamic region selection.

3.2.1 Design choices

We explored a variety of choices while designing Pyramid R-CNN. We tried using a single scale or a pyramid with 7 levels each separated by a scale factor of $2^{-1/2}$. For warping, we experimented with canonical sizes of $s \times s$, for $s \in \{5, 6, 7\}$ and resampling with nearest neighbor or bilinear interpolation. We also tried two variants of the feature pyramid: the raw feature pyramid or one where each level is max pooled with a 3×3 pooling window run at a stride of 1 (to avoid subsampling).

Table 1 shows mAP performance on PASCAL VOC 2007 test for these various choices. The best configuration, in terms of mAP, uses a 7 level pyramid, a warp size of 7×7 with bilinear interpolation, and max pooling. The first level of our pyramid is computed from a 1713×1713 pixel image, which yields a 108×108 cell feature map. The gold standard (non-pyramid) R-CNN performance using the same non-fine-tuned CNN is 44.2% mAP. Our best result with Pyramid R-CNN is slightly worse, at 41.9%.

To map a ROI R into the pyramid, we start by computing an "optimal" scale defined by $\alpha^* = 227/\min(h, w)$, where h and w are image height and width of R. We then find the nearest level in the pyramid: $l^* = \text{argmin}_l |\log(\alpha_l) - \log(\alpha^*)|$, where α_l is the scale factor for pyramid level l. This procedure approximates the scale that R-CNN would use, since it operates on 227 pixel inputs.

- **todo**: Add table

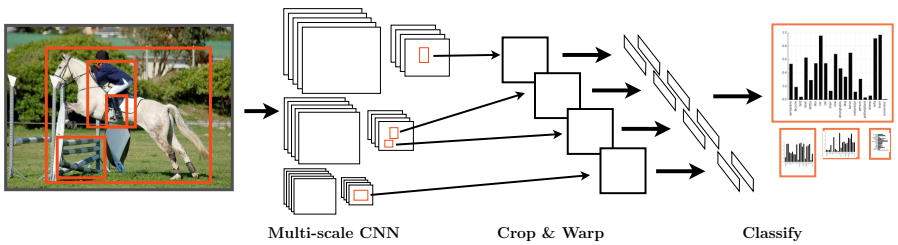

Figure 3: Post R-CNN architecture: the whole image is fed through a CNN up to the highest pooling layer. Regions are cropped from that layer at the best matching scale, resized, and classified.

3.3 Cascaded CNN

Most ROIs that go through the CNN in R-CNN do not contain objects of interest, and are simply background. It would be useful to quickly reject these regions, without expending the full amount of computation on them.

The Cascaded CNN, shown in Figure 4, augments the CNN network with an "Early Terminate" option: after some layers, the network decides whether to keep computing the input with the next layer, or to terminate with a classification. For our purpose, the terminate layers are trained to separate Background/Foreground, terminating if they are confident that the input is Background. Only the last classification layer output the full multi-class scores.

- **todo**: Explain training the thresholds.

Table 1: Pyramid R-CNN design choices.

settings	warp 7x7 nearest	warp 7x7 nearest	warp 7x7 bilinear	max pooled 3x3 warp 7x7 bilinear
scales	1	7	7	7
aeroplane	31.7	46.8	44.5	47.0
bicycle	36.1	50.1	52.6	57.9
bird	12.3	25.4	26.7	31.1
boat	12.5	20.4	25.7	27.5
bottle	11.4	12.9	12.6	19.8
bus	17.1	43.6	45.4	50.7
car	39.8	54.3	56.7	60.0
cat	7.0	39.0	42.4	48.2
chair	6.7	11.7	11.8	16.0
cow	25.8	41.9	44.1	50.1
diningtable	10.2	29.2	33.0	38.6
dog	7.2	35.3	36.0	41.6
horse	17.3	46.3	51.2	55.8
motorbike	27.8	49.4	53.3	56.6
person	16.6	28.8	31.3	36.4
pottedplant	11.4	17.1	18.6	20.9
sheep	19.3	34.4	35.8	40.2
sofa	10.3	22.8	29.3	35.8
train	17.9	38.1	43.0	44.9
tvmonitor	36.3	50.0	54.0	59.7
mAP	18.7	34.9	37.4	41.9

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

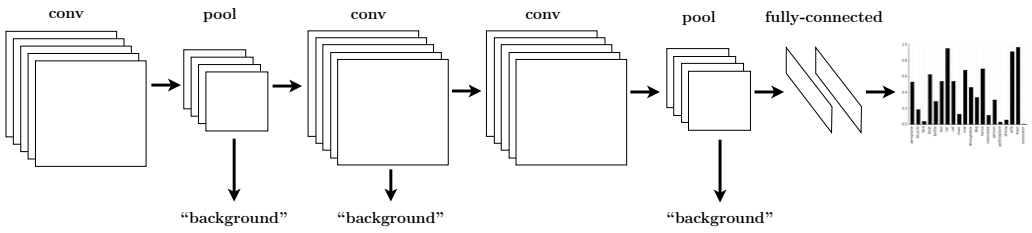

Figure 4: The Cascaded CNN has a reject option after certain layers. **todo**: Redo figure to introduce the Reject layers and color their arrows red to make it clear that a choice is being made.

3.4 Dynamic region selection

Figure 2 shows the dynamic region selection loop.

- **todo**: Explain dynamic selection of region batches.
- **todo**: Explain iterative training procedure.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

4 Evaluation

All results are on PASCAL VOC 2007 and 2010/2012.

- **todo**: TABLE: Dense region evaluation, timing vs. accuracy: effect of multi scale, pooling.
- From the above, we select the point of high accuracy and reasonable cost.
- **todo**: Cascaded CNN timing vs. accuracy: effect of threshold.
- From the above, we select threshold of high accuracy and reasonable cost.
- **todo**: TABLE: Combined system timing vs accuracy: effect of dynamic region selection.
- **todo**: FIGURE: AP vs time plots

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

5 Conclusion

todo: Standard stuff

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

5.1 Future work

- **todo**: ILSVRC results
- **todo**: Training termination layers jointly with the rest of the network.

References

[1] Pierre Sermanet and David Eigen. OverFeat: Integrated Recognition , Localization and Detection using Convolutional Networks. In *ICLR*, 2014.

[2] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[3] J R R Uijlings, K E A Van De Sande, T Gevers, and A W M Smeulders. Selective Search for Object Recognition. *IJCV*, 2013.

[4] Sergey Karayev, Tobias Baumgartner, Mario Fritz, and Trevor Darrell. Timely Object Recognition. In *NIPS*, 2012.

[5] Alex Krizhevsky, Ilyaskever Sut, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*, 2012.

[6] Will Y Zou, Xiaoyu Wang, Miao Sun, and Yuanqing Lin. Generic Object Detection with Dense Neural Patterns and Regionlets. Technical report, 2014.

[7] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep Inside Convolutional Networks : Visualising Image Classification Models and Saliency Maps. In *ICLR*, 2014.

[8] Paul Viola, One Microsoft Way, and Michael J Jones. Robust Real-Time Face Detection. *IJCV*, 57(2):137–154, 2004.

[9] Pedro F Felzenszwalb, Ross B Girshick, and David McAllester. Cascade object detection with deformable part models. In *CVPR*, pages 2241–2248. IEEE, June 2010.

[10] Piotr Dollar, Ron Appel, and Wolf Kienzle. Crosstalk Cascades for Frame-Rate Pedestrian Detection. In *ECCV*, 2012.