Deformable Part Models are Convolutional Neural Networks

Xuewen Yang

August 15 2018

1. Implementation details

The author implemented their experiments by modifying the DPM voc-release code and using Caffe for CNN computation.They describe the most important implementation details. Source code for the complete system is available, thus providing documentation of the remaining implementation details.

1.1. Parameter learning

There are at least two natural ways to train a DeepPyramid DPM. The first treats the model as a single CNN and trains it end-to-end with SGD and backpropagation. The second trains the model in two stages: (1) fit the front-end CNN; (2) train a DPM on top of stage 1 using latent SVM [1] while keeping the front-end CNN fixed. The first procedure is more in the spirit of deep learning, while the second is an important baseline for comparison to traditional HOG-based DPM and for showing if end-to-end training is useful.They chose to focus on latent SVM training since it is a necessary baseline for end-to-end training. Although we dont explore end-to-end training due to space constraints, they point interested readers to contemporaneous work by Wan *et al.* that shows results of a similar model trained end-to-end. They report modest improvements from endto- end optimization.

1.2. Feature pyramid construction

Any fully-convolutional network can be used to generate the feature pyramid. In this DeepPyramid DPM implementation,they chose to use a truncated variant of the SuperVision CNN [2]. In order to directly compare our results with R-CNN, they use the publicly available network weights that are distributed with R-CNN. These weights were trained on the ILSVRC 2012 classification training dataset using Caffe (they do not use the detection fine-tuned weights since they were trained on warped image windows).

1.3. DPM training and testing details

Compared to training a DPM with HOG features, they found it necessary to make some changes to the standard DPM training procedure. First, they dont use left/right mirrored pairs of mixture components. These components are easy to implement with HOG because model parameters can be explicitly flipped allowing them to be tied between mirrored components. Second, R-CNN and DPM use different non-maximum suppression functions and they found that the one used in R-CNN, which is based on intersectionover- union (IoU) overlap, performs slightly better with conv5 features (but is worse for the baseline HOG-DPM). Lastly, they found that its very important to use poorly localized detections of ground-truth objects as negative examples. As in R-CNN, they define negative examples as all detections that have a max IoU overlap with a ground-truth object of less than 0.3. Using poorly localized positives as negative examples leads to substantially better results than just using negatives from negative images, which is the standard practice when training HOG-DPM. Using these difficult negative examples in HOG-DPM did not improve the baseline results.

References

[1] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010. 1

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1