Supplemental Material: Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks

Sean Bell[1] C. Lawrence Zitnick[2] Kavita Bala[1] Ross Girshick[2]
[1]Cornell University [2]Microsoft Research*
{sbell,kb}@cs.cornell.edu {rbg,zitnick}@fb.com

2015 MS COCO Competition Submission

In this section, we describe our submission to the 2015 MS COCO Detection Challenge, which won Best Student Entry and finished 3rd place overall, with a score of 31.0% mAP on 2015 test-challenge and 31.2% on 2015 test-dev. Later in this section we describe further post-competition improvements to achieve 33.1% on test-dev. Both models use a single ConvNet (no ensembling).

For our challenge submission, we made several improvements: used a mix of MCG (Multiscale Combinatorial Grouping [3]) and RPN (Region Proposal Net [4]) proposal boxes, added two extra 512x3x3 convolutional layers, trained for longer, and used two rounds of bounding box regression with a modified version of weighted voting [1]. At test time, our model runs in 2.7 seconds/image on a single Titan X GPU (excluding proposal generation).

We describe all changes in more detail below (note that many of these choices were driven by the need to meet the challenge deadline, and thus may be suboptimal):

1. **Train+val.** For the competition, we train on both train and validation sets. We hold out 5000 images from the validation as our new validation set called "minival."
2. **MCG+RPN box proposals.** We get the largest improvement by replacing selective search with a mix of MCG [3] and RPN [4] boxes. We modify RPN from the baseline configuration described in [4] by adding more anchor boxes, in particular smaller ones, and using a mixture of 3x3 (384) and 5x5 (128) convolutions. Our anchor configuration uses a total of 22 anchors per location with the following shapes: 32x32 and aspect ratios {1:2, 1:1, 2:1} $\times$ scales {64, 90.5, 128, 181, 256, 362, 512}. We also experiment with predicting proposals from concatenated conv4_3 and conv5_3 features (after L2 normalization and scaling), rather than from conv5_3 only. We refer to these two configurations as RPN1 (without concatenated features) and RPN2 (with concatenated features). Using VGG16, RPN1 and RPN2 achieve average recalls of 44.1% and 44.3%, respectively, compared to selective search, 41.7%, and MCG, 51.6%. Despite their lower average recalls, we found that RPN1 and RPN2 give comparable detection results to MCG. We also found that mixing 1000 MCG boxes with 1000 RPN1 or RPN2 boxes performs even better than 2000 of either method alone, suggesting that the two methods are complementary. Thus, we train with 1000 RPN1 and 1000 MCG boxes. At test time, we use 1000 RPN2 and 2000 MCG boxes, which gives a +0.3 mAP improvement on minival compared to using the same boxes as training.
3. **conv6+conv7.** We use the model listed in row (c) of Table 9: two 512x3x3 convolutions on top of conv5_3 which we call "conv6" and "conv7". We also pool out of conv7, so in total we pool out of conv3_3, conv4_3, conv5_3, conv7, and IRNN.
4. **Longer training.** Since training on COCO is very slow, we explored ideas by initializing new networks from the weights of the previous best network. While we have not tried training our best configuration starting from VGG16 ImageNet weights, we do not see any reason why it would not work. Here is the exact training procedure we used:
   (a) Train using 2000 selective search boxes, using the schedule described in Section 4.1, but stopping after 220k iterations of fine-tuning. This achieves 24.8% on 2015 test-dev.
   (b) Change the box proposals to a mix of 1000 MCG boxes and 1000 RPN1 boxes. Train for 100k iterations with conv layers frozen (learning rate: exp decay $5 \cdot 10^{-3} \rightarrow 10^{-4}$), and for 80k iterations with only conv1 and conv2 frozen (learning rate: exp decay $10^{-3} \rightarrow 10^{-5}$).
5. **2xBBReg + WV.** We use the iterative bounding box regression and weighted voting scheme described in [1], but as noted in Section 4.4, this does not work out-of-the-box for COCO since boxes are blurred together and precise localization is lost. We solve this by adjusting the thresholds so that only very similar boxes are blurred together with weighted voting. We jointly

*R. Girshick and C. L. Zitnick are now at Facebook AI Research.

Method	Proposals	Train: / Test:	train minival	train test-dev	trainval35k minival	trainval35k test-dev
baseline VGG16	Sel. Search		20.3	20.5		
ION VGG16	Sel. Search		24.4	24.9		
ION VGG16 + conv7	Sel. Search			25.1		
ION VGG16 + conv7	MCG + RPN				28.4	29.0
+ **W** (box refinement)					30.0	30.6
+ flipping + more, better training					32.5	**33.1**

Table 1. Breakdown of gains for the post-competition model. The reported metric is Avg. Precision, IoU: 0.5:0.95. The training set "trainval35k" includes all of train together with approximately 35k images from val, after removing the 5k minival set. All entries use a single ConvNet model (no ensembling). The majority of the gains come from the ION model (20.5 $\rightarrow$ 24.9) and better proposals with more training data (MCG + RPN: 25.1 $\rightarrow$ 29.0). Two rounds of bounding box regression with weighted voting and longer training with improved hyperparameters also yield important gains. Note that we use a modified version of RPN [4], described in the Supplemental text.

optimized the NMS (non-max suppression) and voting threshold on the validation set, by evaluating all boxes once and then randomly sampling hundreds of thresholds. On our minival set, the optimal IoU (intersection-over-union) threshold is 0.443 for NMS and 0.854 for weighted voting. Compared to using a single round of standard NMS (IoU threshold 0.3), these settings give +1.3 mAP on minival.

Post-competition improvements

We have an improved model which did not finish in time for the challenge, which achieves 33.1% on 2015 test-dev. We made these further adjustments:

1. **Training longer with 0.99 momentum.** We found that if the momentum vector is reset when re-starting training, the accuracy drops by several points and takes around 50k iterations to recover. Based on this observation, we increased momentum to 0.99 and correspondingly decreased learning rate by a factor of 10. We initialized from our competition submission (above), and:
   (a) further trained for 80k iterations with only conv1 and conv2 frozen, (learning rate: exp decay $10^{-4} \rightarrow 10^{-6}$). By itself, this gives a boost of +0.4 mAP on minival.
   (b) We trained for another 160k iterations with no layers frozen (learning rate: exp decay $10^{-4} \rightarrow 10^{-6}$) which gives +0.7 mAP on minival.
2. **Left-right flipping.** We evaluate the model twice, once on the original image, and once on the left-right flipped image. To merge the predictions, we average both the softmax scores and box regression shifts (after flipping back). By itself, this gives a boost of +0.8 mAP on minival.
3. **More box proposals.** At test time, we use 4000 proposal boxes (2000 RPN2 and 2000 MCG boxes). For the model submitted to the competition, this performs +0.1 mAP better than 3000 boxes (1000 RPN2 and 2000 MCG boxes).

At test time, the above model runs in 5.5 seconds/image on a single Titan X GPU (excluding proposal generation). Most of the slowdown is from the left-right flipping. Table 1 provides a breakdown of the gains due to the various competition and post-competition changes.

Visual results for COCO 2015

To give a sense for the performance of our method, we include some qualitative results for COCO 2015 test-dev:

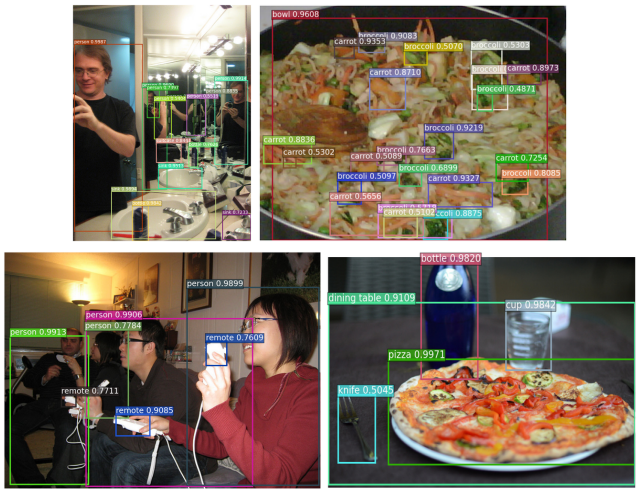

Figure 1. **Challenging detections** on COCO 2015 test-dev using our model trained on COCO "2014 train."

Additional Evaluation (on VOC 2007)

In this section, we include more PASCAL VOC 2007 test results, that did not fit in the main paper.

Number of IRNN layers

As mentioned in the main paper, using 2 IRNN layers performs the best on VOC 2007 test. Table 2 shows the

breakdown in performance for different number of layers:

ROI pooling from:					Seg.	# IRNN layers		
C2	C3	C4	C5	IRNN		1	2	3
				✓	✓		70.6	
			✓	✓	✓	74.3		
		✓	✓	✓	✓	75.8	76.2	
	✓	✓	✓	✓	✓	76.1	76.5	75.9
✓	✓	✓	✓	✓	✓		**76.8**	

Table 2. **Varying the number of IRNN layers.** Metric: mAP on VOC07 test. Segmentation loss is used to regularize the top IRNN layer. All IRNNs use 512 hidden units.

Visualizing IRNN hidden state

To gain more insight into the IRNN layers, we directly visualize their hidden state in Figure 2. We can see that the hidden state is slowly varying, and thus can encode more interesting computation than simple global average pooling.

Additional plots for VOC 2007 error diagnosis

In this section, we include the full error analysis for ION (our object detector), tested on PASCAL VOC 2007 test, trained on 2007 trainval + 2012 trainval. These plots are from Hoiem's toolkit for diagnosing errors [2]. Figure 4 in the main paper is a condensed version of these plots, focusing only on area.

From these plots, we can see that we see large gains across the board, but particularly for small objects.

References

[1] S. Gidaris and N. Komodakis. Object detection via a multi-region & semantic segmentation-aware CNN model. In *ICCV*, 2015. 1

[2] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing error in object detectors. In *ECCV*, 2012. 3

[3] J. Pont-Tuset, P. Arbeláez, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping for image segmentation and object proposal generation. In *arXiv:1503.00848*, March 2015. 1

[4] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 1, 2

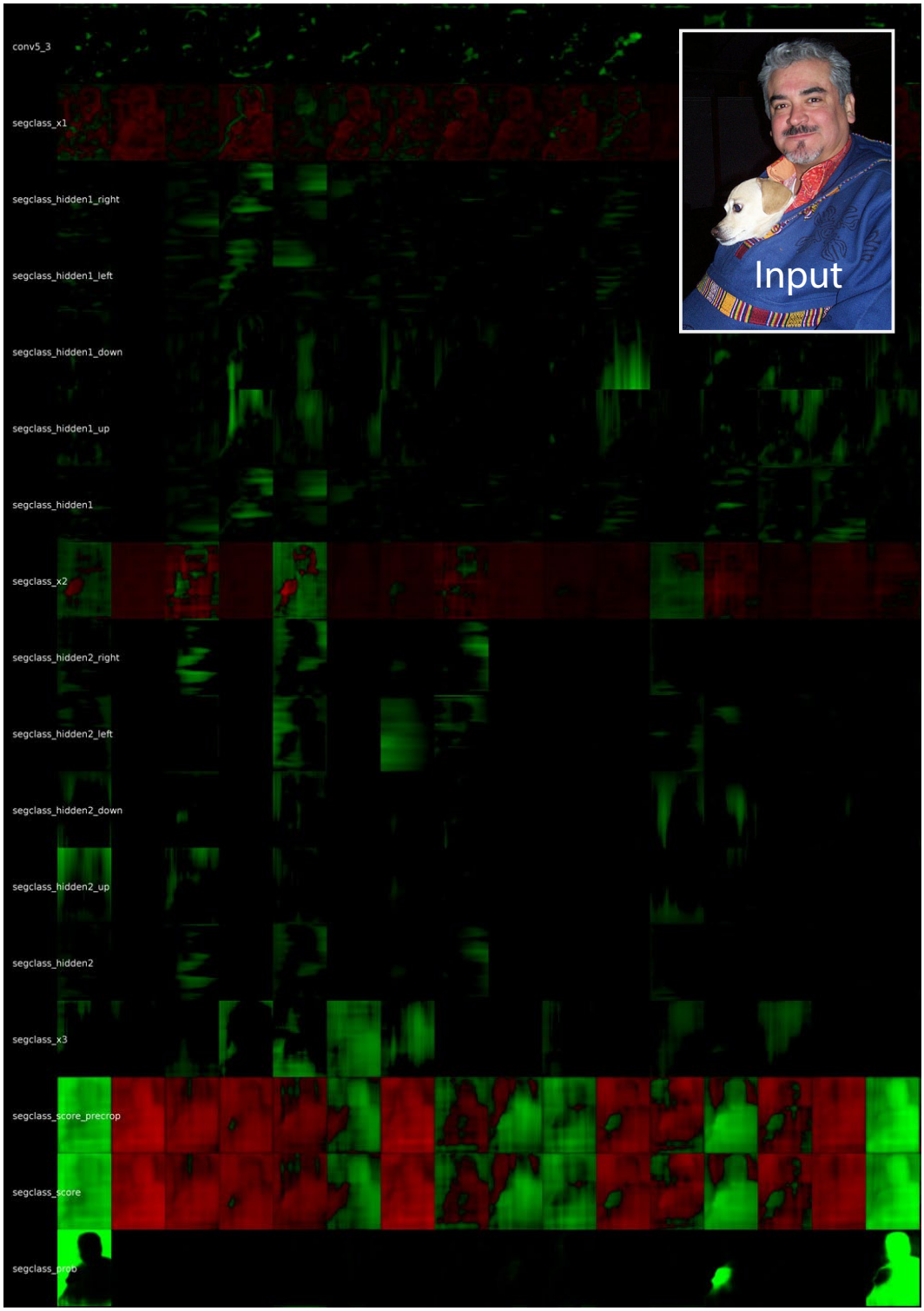

Figure 2. **Visualizing hidden state** for a VOC2007 test image. Green corresponds to positive values, and red corresponds to negative. Each row visualizes the output from different layer in the network, and each column is a different channel for that layer's output. "segclass_x1": the first input-to-hidden 1x1 convolution, "segclass_hidden1_right": the hidden state of the first hidden-to-hidden recurrence (similarly for other directions), "segclass_hidden1": after concatenating the hidden state from the 4 layers, "segclass_x2": the second input-to-hidden 1x1 convolution, "segclass_hidden2_right": the second layer of the 4-dir IRNN hidden states, "segclass_x3": the final output of the stacked IRNN (a 1x1 convolution), "segclass_score": the predicted segmentation labels (used for regularization only).

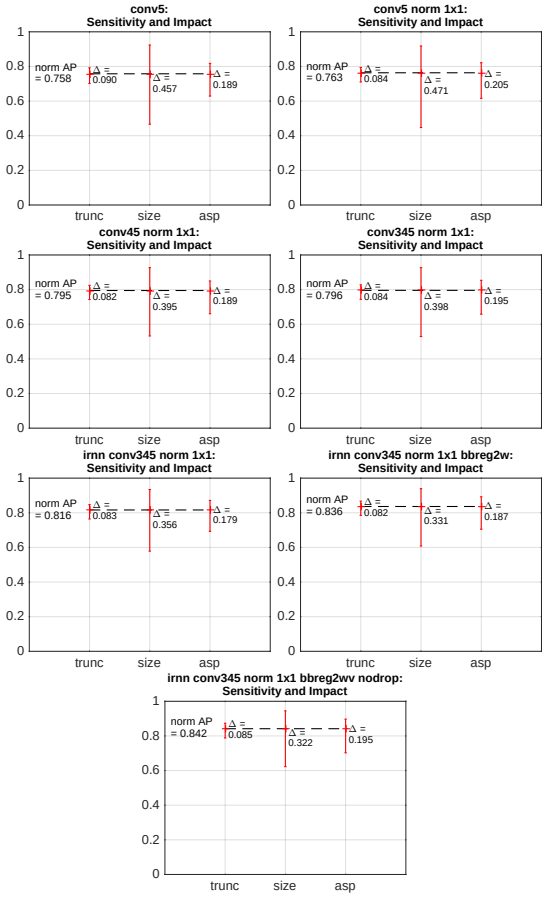

Figure 3. **Impact.** We can see that as our model complexity increases, the impact due to size decreases.

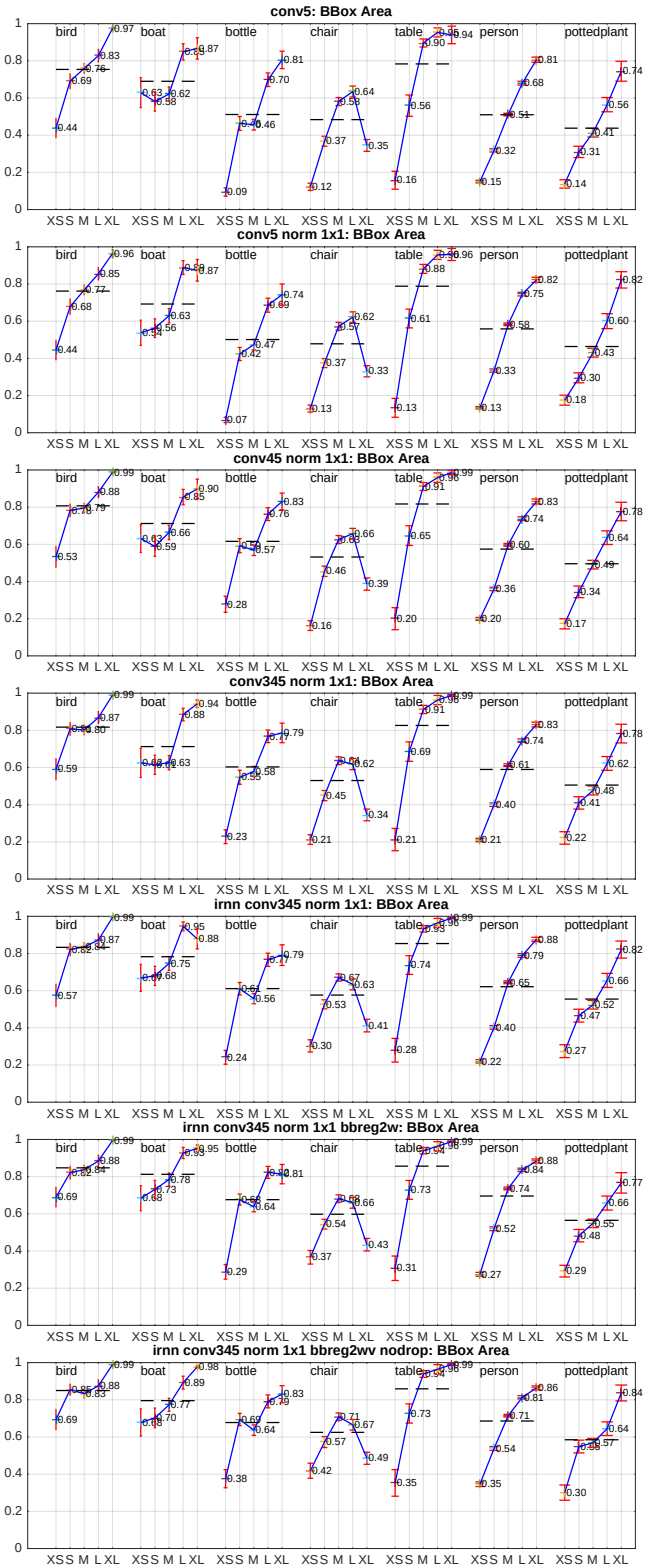

Figure 4. **VOC 2007 normalized AP by area.**

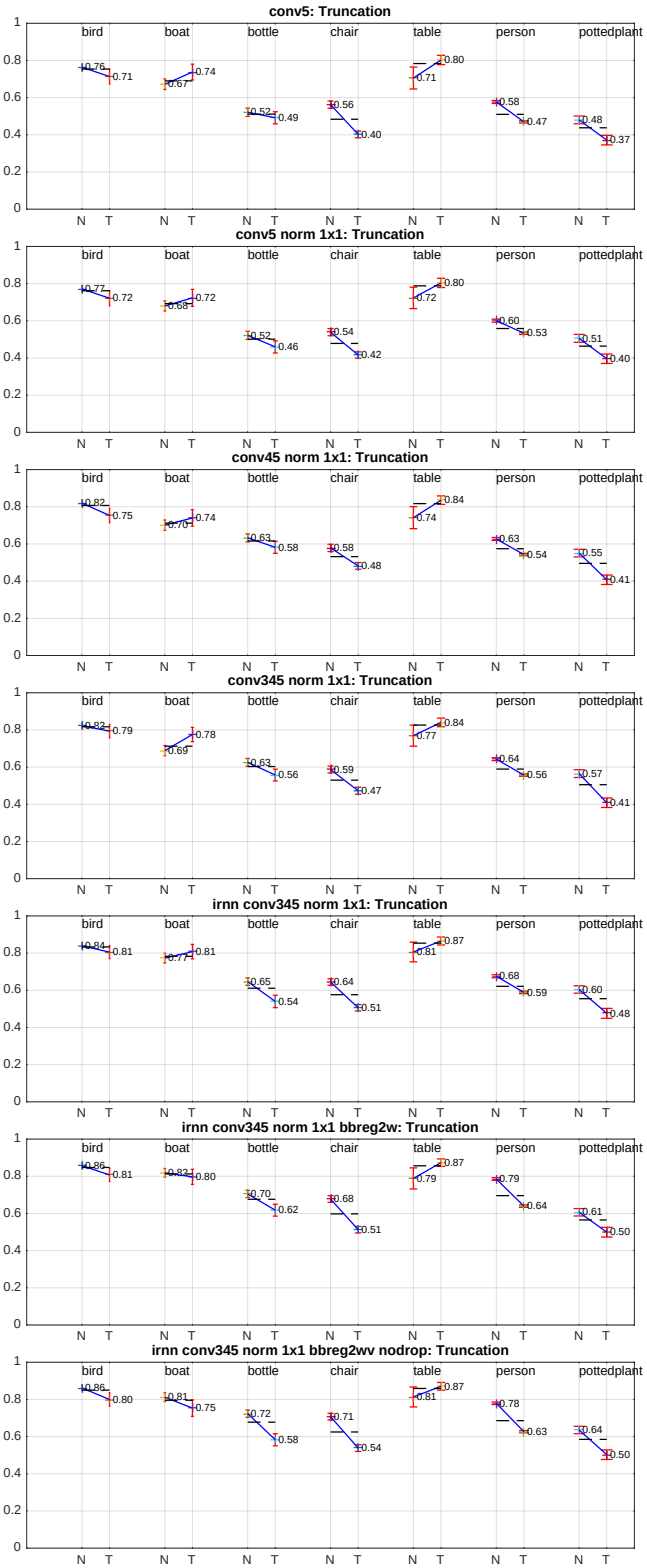

Figure 5. **VOC 2007 normalized AP by truncation.** N: not truncated, T: truncated.

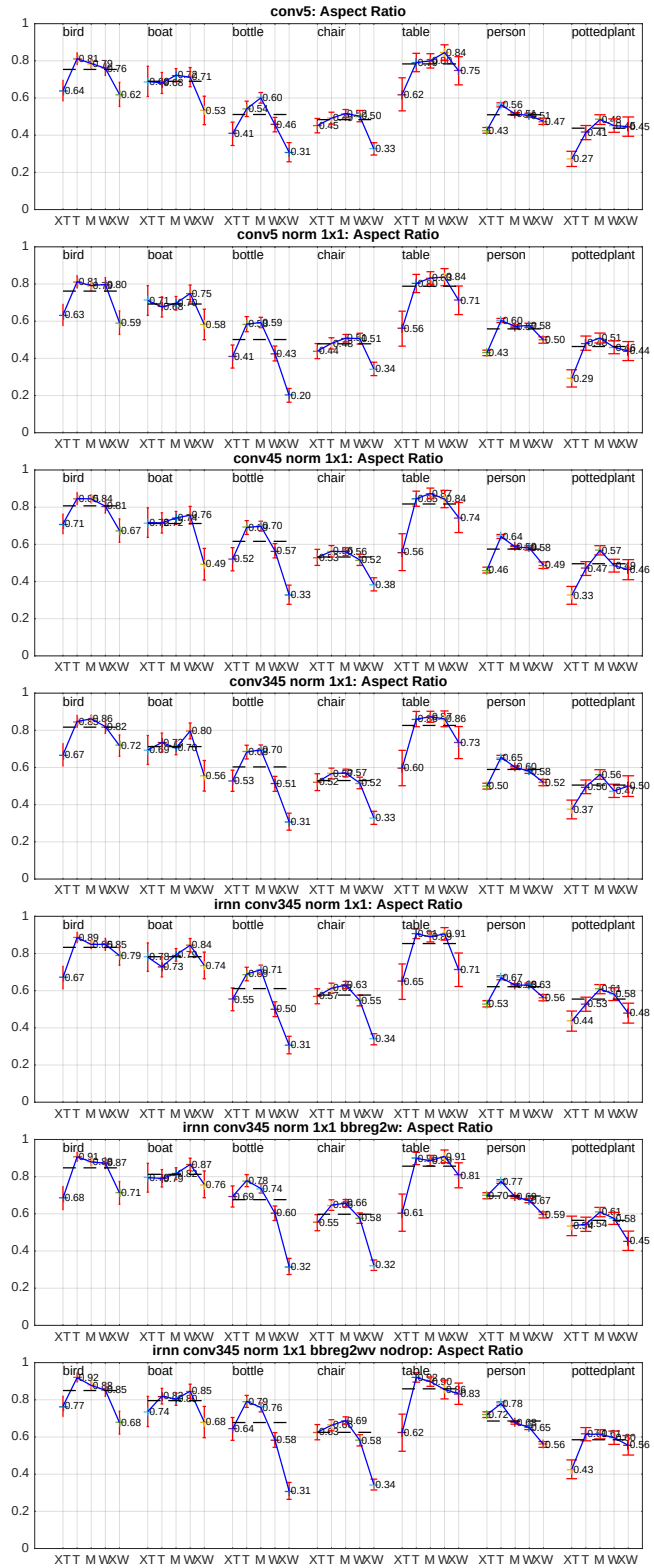

Figure 6. aspect

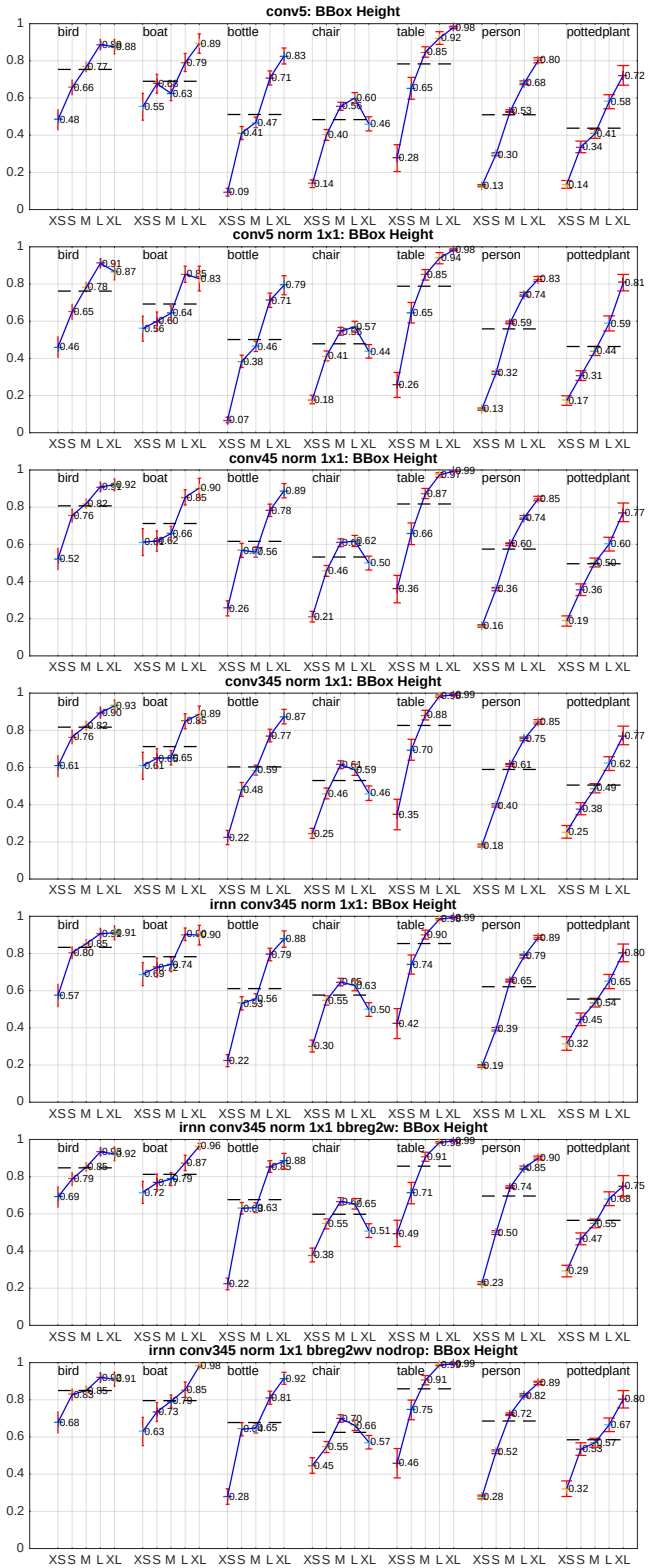

Figure 7. **VOC 2007 normalized AP by height.**

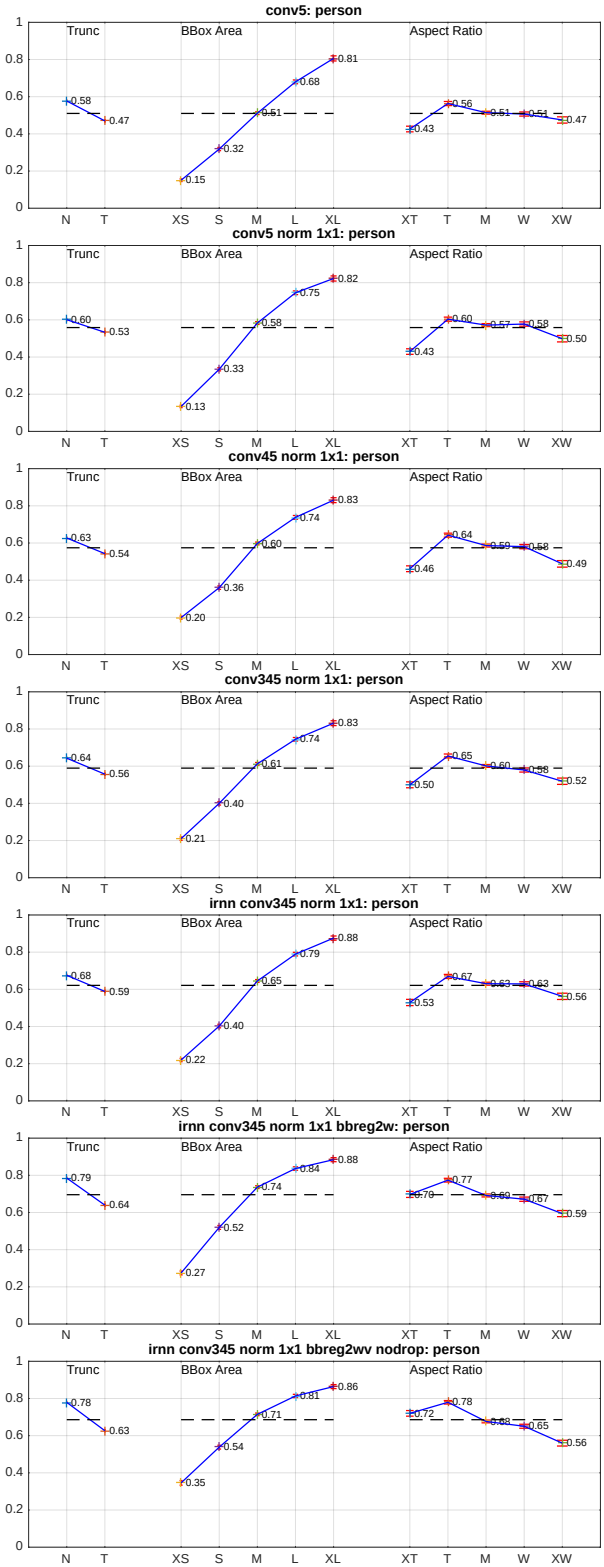

Figure 8. **VOC 2007 normalized AP for "person."**

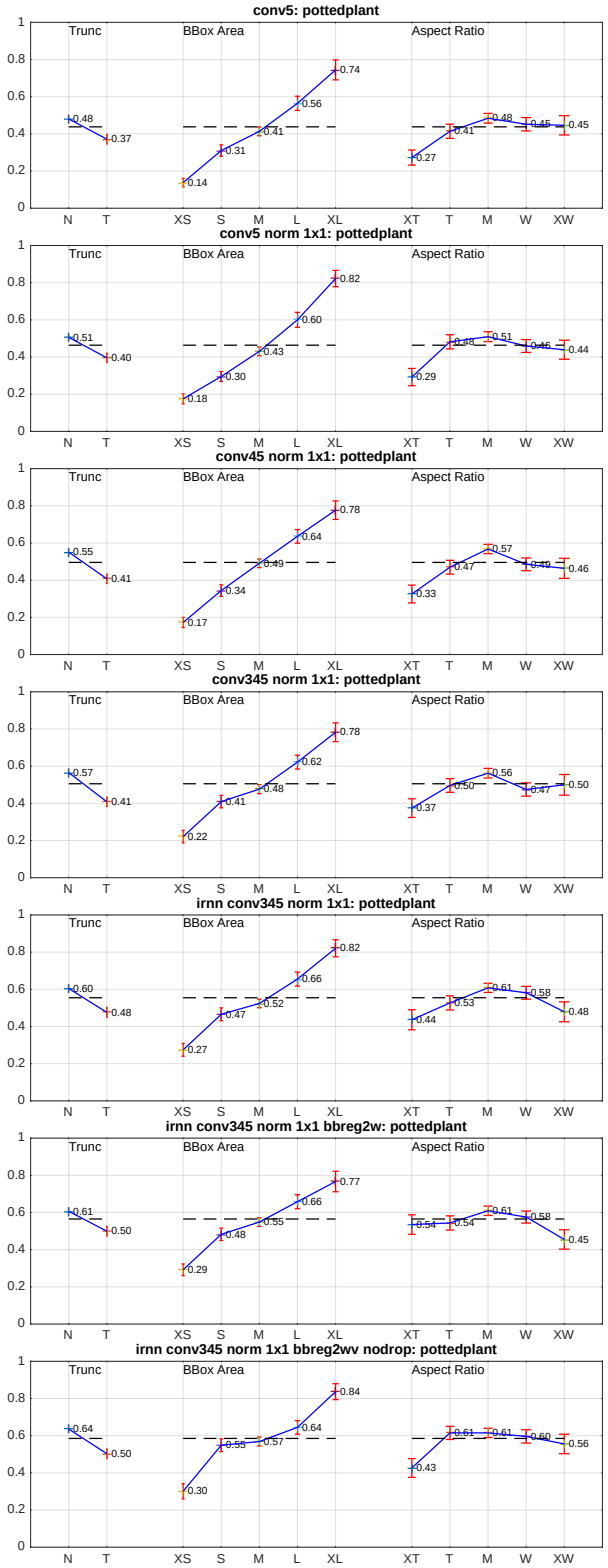

Figure 9. pottedplant

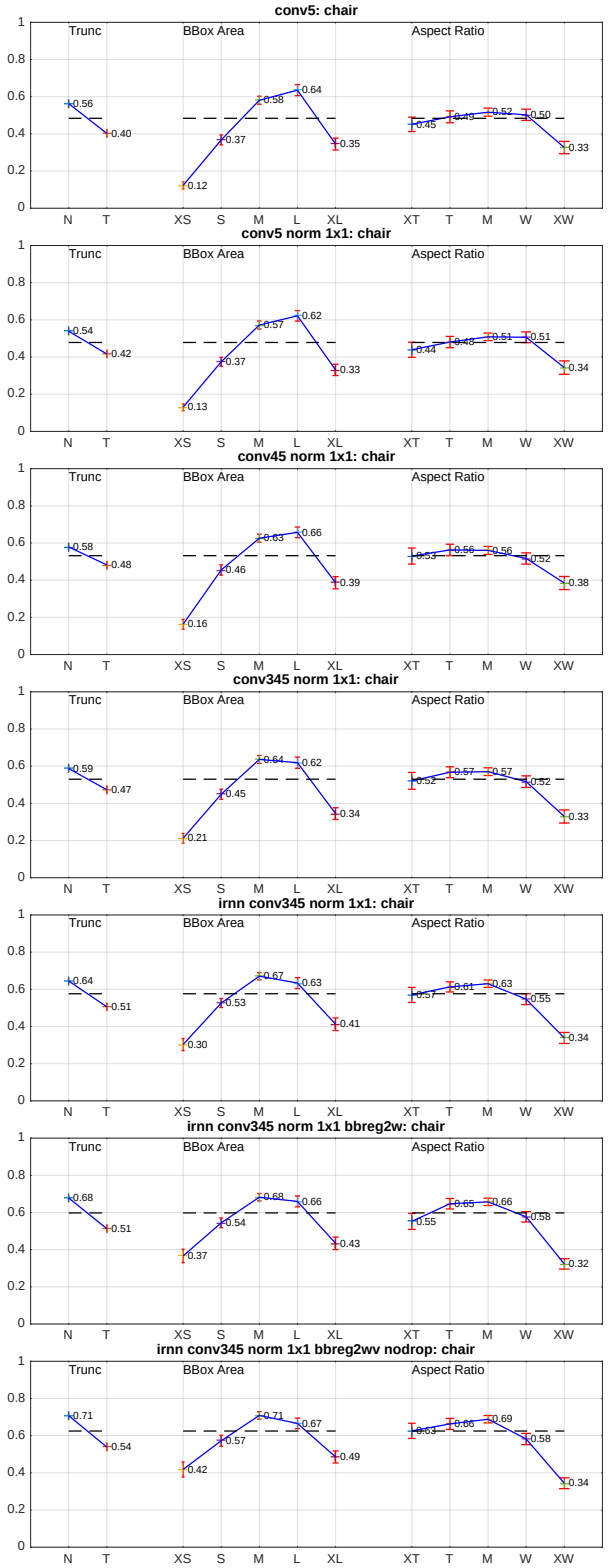

Figure 10. **VOC 2007 normalized AP for "chair."**

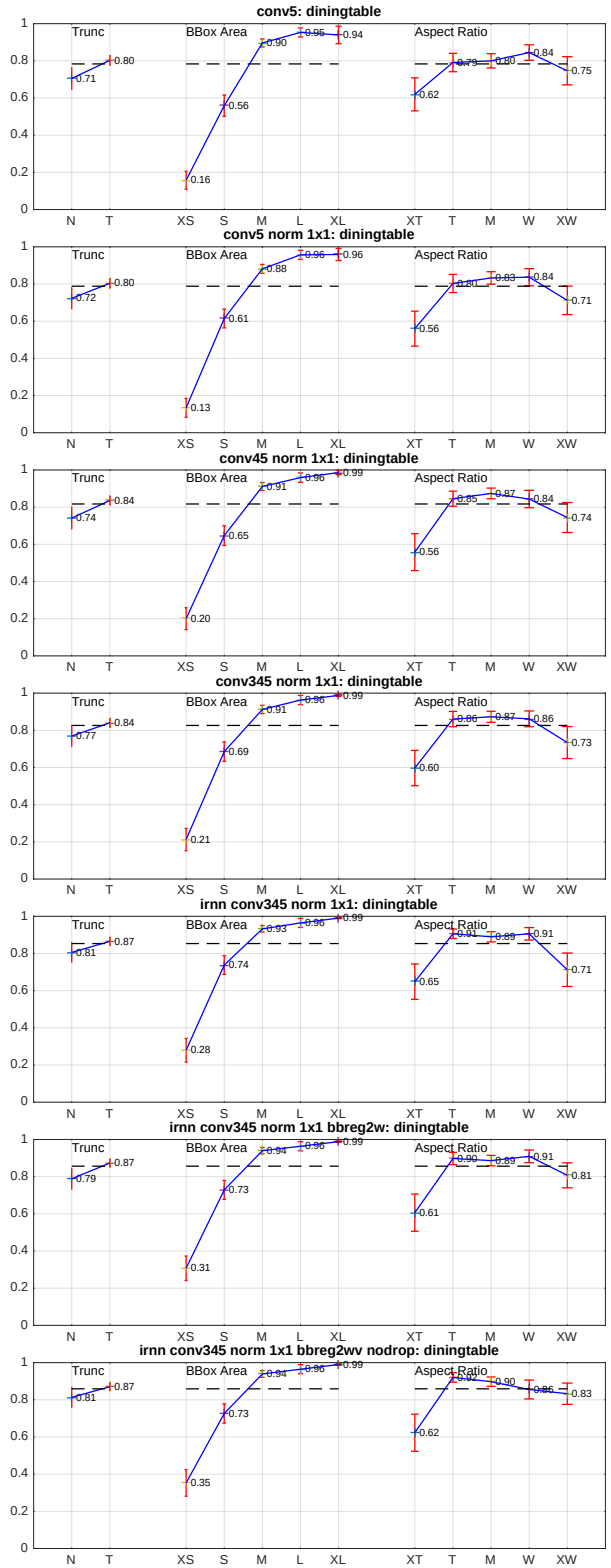

Figure 11. **VOC 2007 normalized AP for "dining table."**

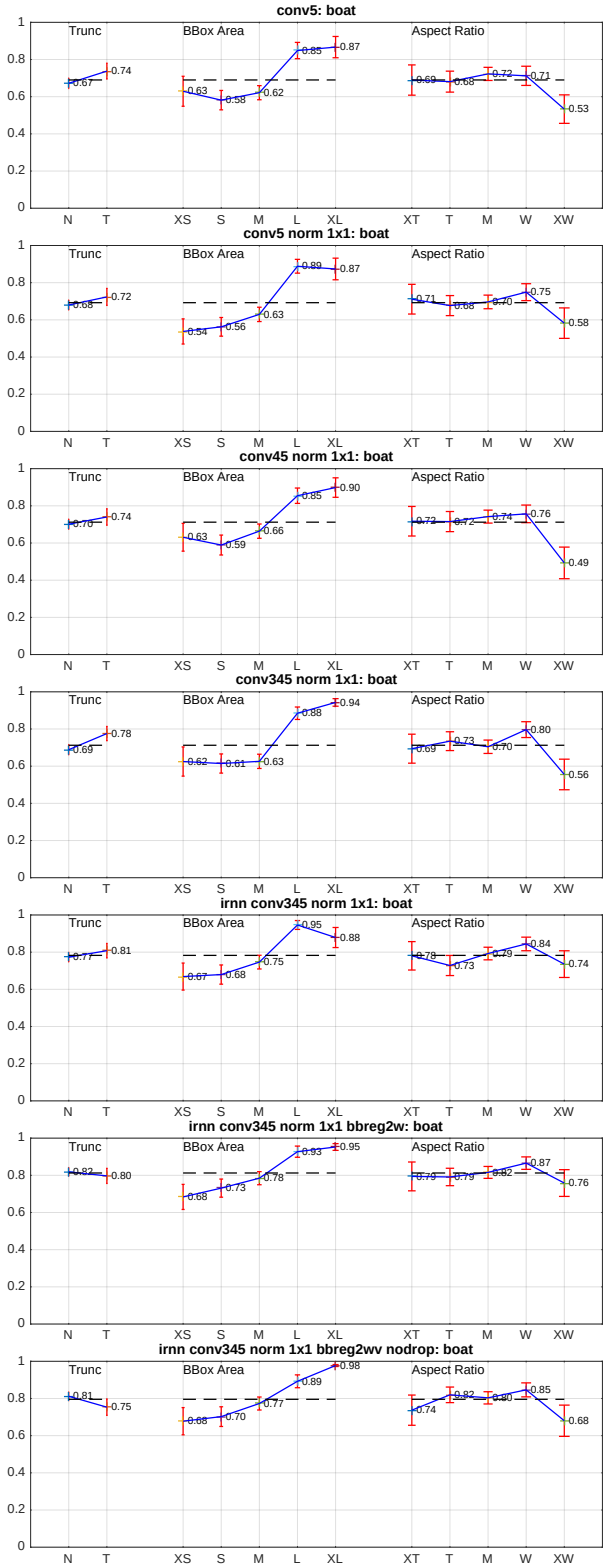

Figure 12. **VOC 2007 normalized AP for "boat."**

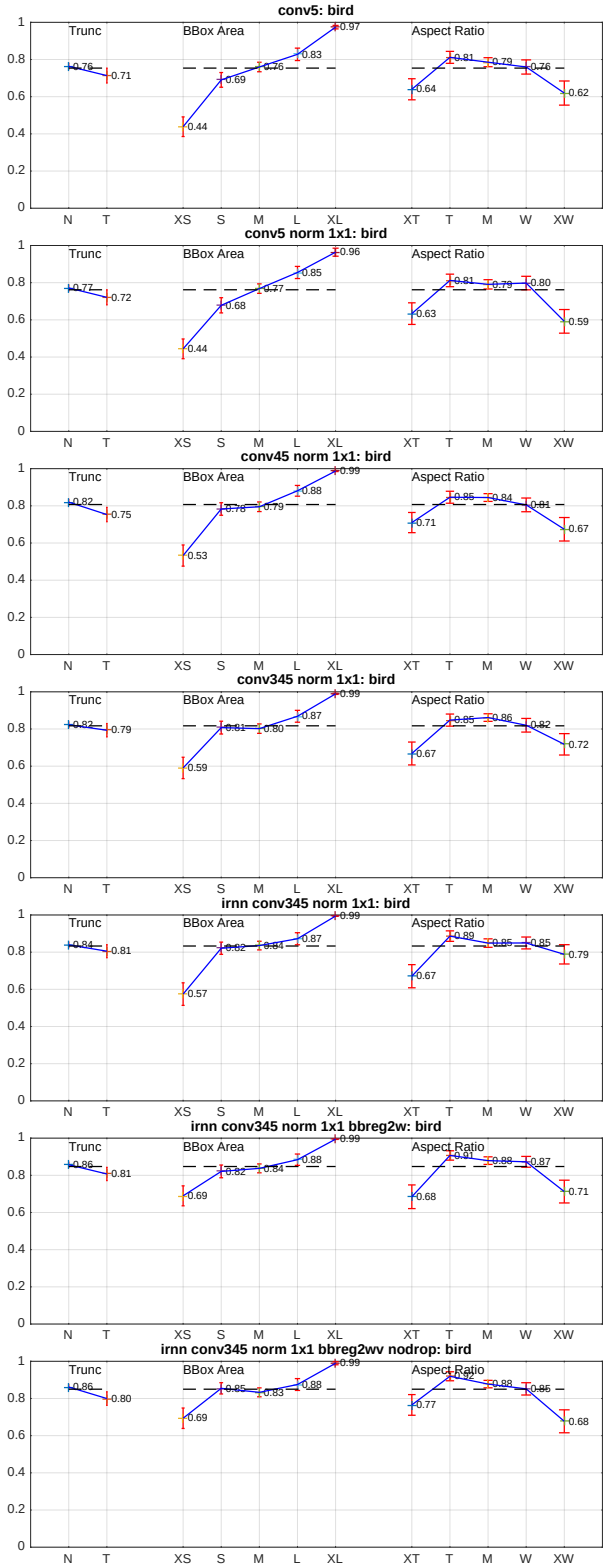

Figure 13. **VOC 2007 normalized AP for "bird."**