Supplementary Material:
Pixelwise Instance Segmentation with a Dynamically Instantiated Network

Anurag Arnab and Philip H.S. Torr
University of Oxford
{anurag.arnab, philip.torr}@eng.ox.ac.uk

1. Introduction

In this supplementary material, we include more detailed qualitative and quantitative results on the VOC and SBD datasets. Furthermore, we also show the runtime of our algorithm.

Figures 1 and 2 show success and failure cases of our algorithm. Figure 3 compares the results of our algorithm to the publicly available model for MNC [3]. Figure 4 compares our results to those of FCIS [6], concurrent work which won the COCO 2016 challenge. Figure 5 presents some qualitative results on the Cityscapes dataset.

Section 2 shows more detailed results on the VOC dataset. Figure 6 shows a visualisation of our results at different AP^r thresholds, and Tables 2 to 4 show per-class AP^r results at thresholds of 0.5, 0.7 and 0.9.

Section 3 shows more detailed results on the SBD dataset. Table 1 shows our mean AP^r results at thresholds from 0.5 to 0.9, whilst Tables 5 and 6 show per-class AP^r results at thresholds of 0.7 and 0.5 respectively.

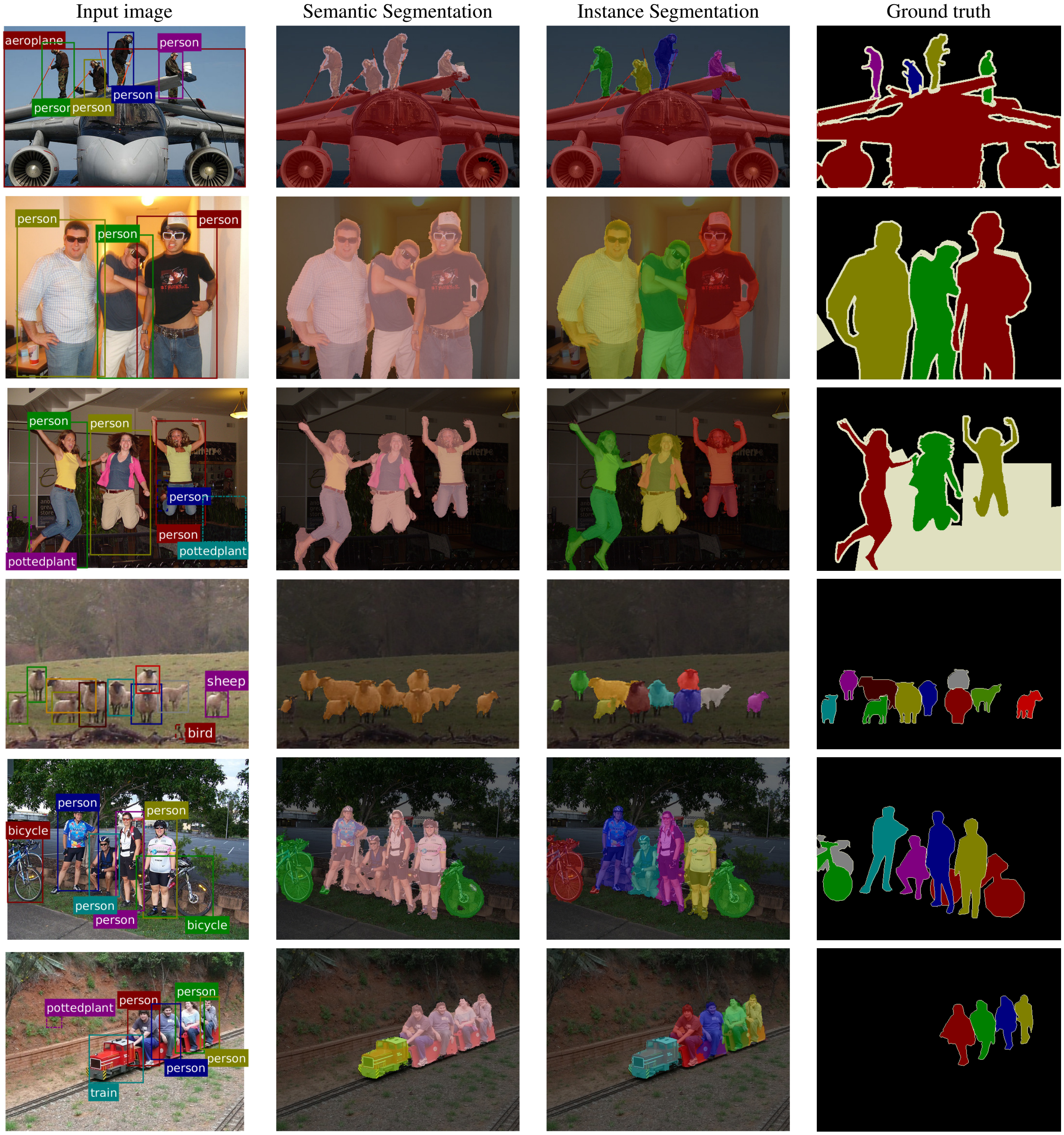

Figure 1: **Success cases of our method.** *First and second row:* Our algorithm can leverage good initial semantic segmentations, and detections, to produce an instance segmentation. *Third row:* Notice that we have ignored three false-positive detections. Additionally, the red bounding box does not completely encompass the person, but our algorithm is still able to associate pixels "outside-the-box" with the correct detection (also applies to row 2). *Fourth row:* Our system is able to deal with the heavily occluded sheep, and ignore the false-positive detection. *Fifth row:* We have not been able to identify one bicycle on the left since it was not detected, but otherwise have performed well. *Sixth row:* Although subjective, the train has not been annotated in the dataset, but both our initial semantic segmentation and object detection networks have identified it. Note that the first three images are from the VOC dataset, and the last three from SBD. Annotations in the VOC dataset are more detailed, and also make more use of the grey "ignore" label to indicate uncertain areas in the image. The first column shows the input image, and the results of our object detector which are another input to our network. Best viewed in colour.

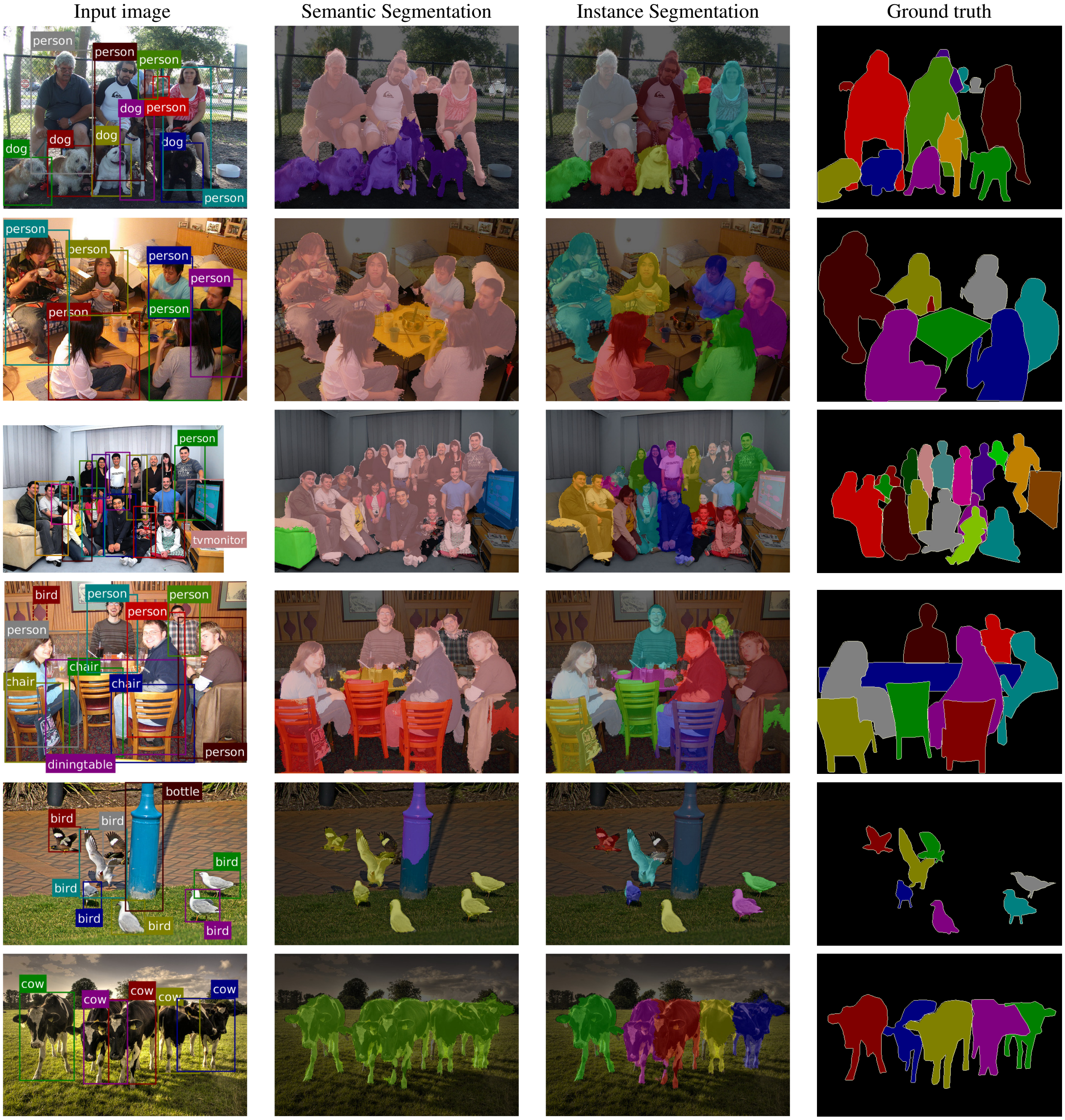

Figure 2: **Failure cases of our method.** *First row:* Both our initial detector, and semantic segmentation system did not identify a car in the background. Additionally, the "brown" person prediction actually consists of two people that have been merged together. This is because the detector did not find the background person. *Second row:* Our initial semantic segmentation identified the table, but it is not there in the Instance Segmentation. This is because there was no "table detection" to associate these pixels with. Using heuristics, we could propose additional detections in cases like these. However, we have not done this in our work. *Third row:* A difficult case where we have segmented most of the people. However, sometimes two people instances are joined together as one person instance. This problem is because we do not have a detection for each person in the image. *Fourth row:* Due to our initial semantic segmentation, we have not been able to segment the green person and table correctly. *Fifth row:* We have failed to segment a bird although it was detected. *Sixth row:* The occluding cows, which all appear similar, pose a challenge, even with our shape priors. The first column shows the input image, and the results of our object detector which are another input to our network. Best viewed in colour.

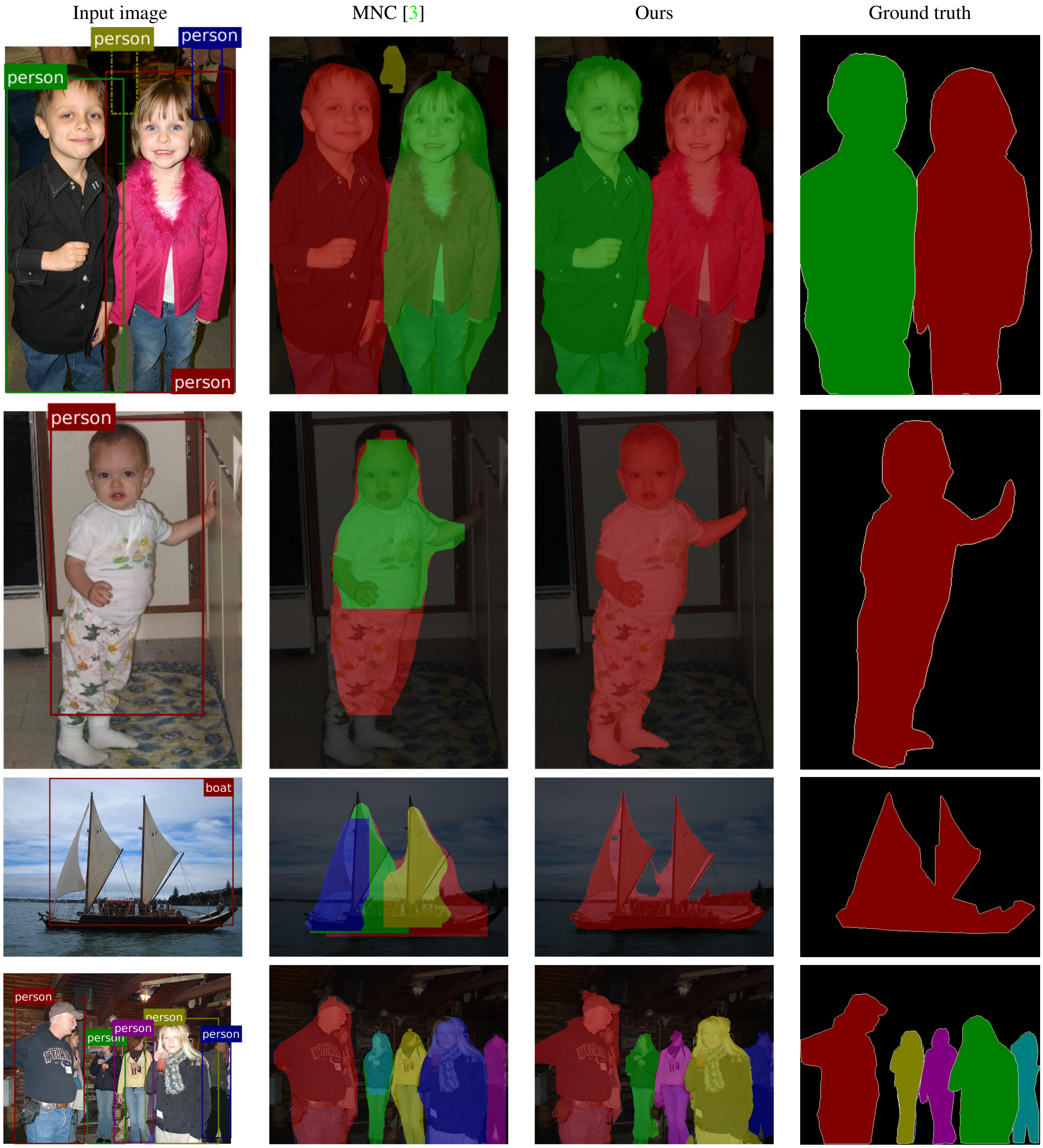

Figure 3: **Comparison to MNC [3]** The above examples emphasise the advantages in our method over MNC [3]. Unlike proposal-based approaches such as MNC, our method can handle false-positive detections, poor bounding box localisation, reasons globally about the image and also produces more precise segmentations due to the initial semantic segmentation module which includes a differentiable CRF. *Row 1* shows a case where MNC, which scores segment-based proposals, is fooled by a false-positive detection and segments an imaginary human (yellow segment). Our method is robust to false-positive detections due to the initial semantic segmentation module which does not have the same failure modes as the detector. *Rows 2, 3 and 4* show how MNC [3] cannot deal with poorly localised bounding boxes. The horizontal boundaries of the red person in Row 2, and light-blue person in Row 4 correspond to the limits of the proposal processed by MNC. Our method, in contrast, can segment "outside the detection bounding box" due to the global instance unary potential (Eq. 4). As MNC does not reason globally about the image, it cannot handle cases of overlapping bounding boxes well, and produces more instances than there actually are. The first column shows the input image, and the results of our object detector which are another input to our network. MNC does not use these detections, but does internally produce box-based proposals which are not shown. Best viewed in colour.

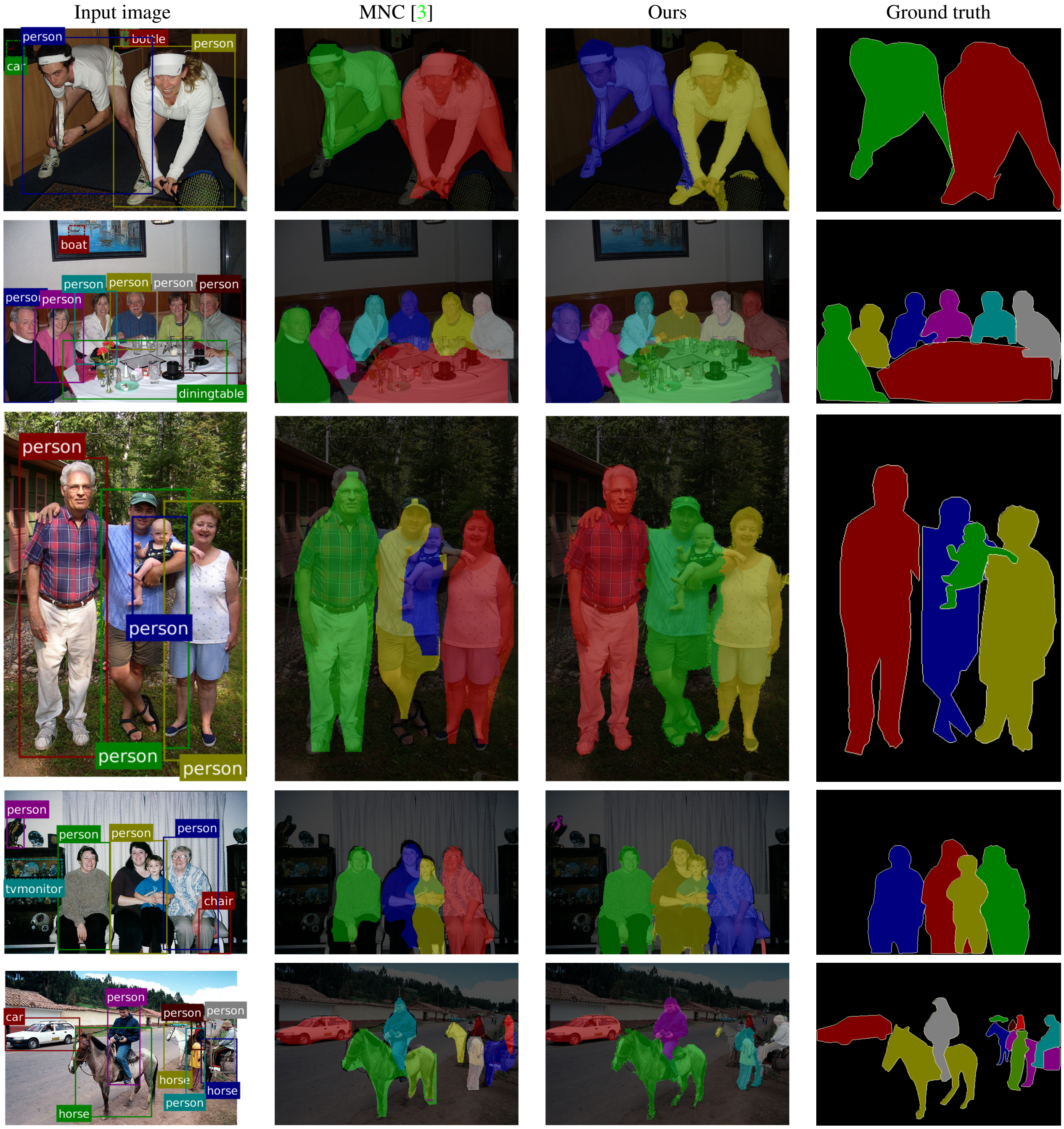

Figure 3 continued: **Comparison to MNC [3]** The above examples show that our method produces more precise segmentations than MNC, that adhere to the boundaries of the objects. However, in Rows 3, 4 and 5, we see that MNC is able to segment instances that our method misses out. In *Row 3*, our algorithm does not segment the baby, although there is a detection for it. This suggests that our shape prior which was formulated to overcome such occlusions could be better. As MNC processes individual instances, it does not have a problem with dealing with small, occluding instances. In *Row 4*, MNC has again identified a person that our algorithm could not. However, this is because we did not have a detection for this person. In *Row 5*, MNC has segmented the horses on the right better than our method. The first column shows the input image, and the results of our object detector which are another input to our network. MNC does not use these detections, but does internally produce box-based proposals which are not shown. We used the publicly available code, models and default parameters of MNC to produce this figure. Best viewed in colour.

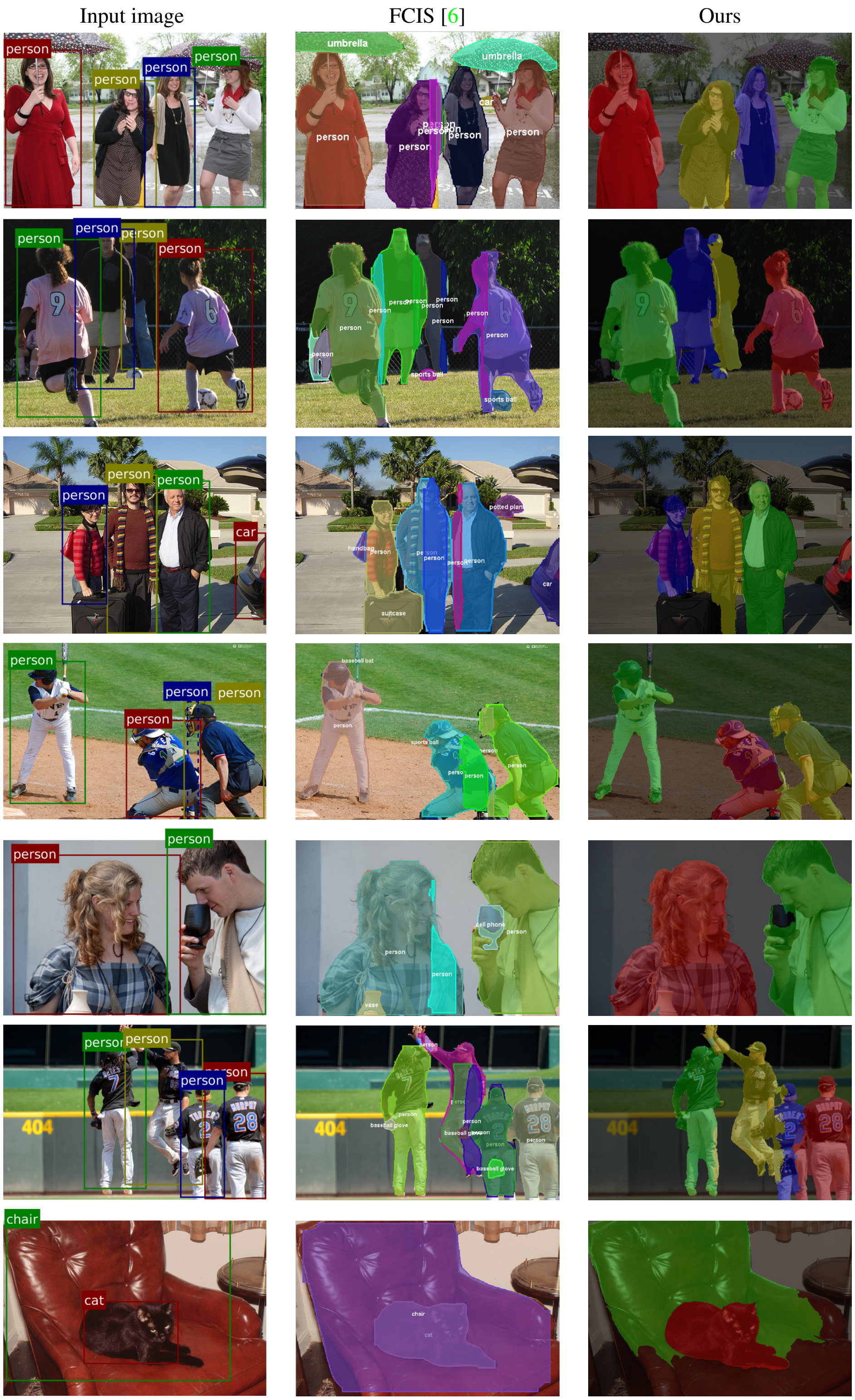

Figure 4: **Comparison to FCIS [6]** The above images compare our method to the concurrent work, FCIS [6], which was trained on COCO [8] and won the COCO 2016 challenge. Unlike proposal-based methods such as FCIS, our method can handle false-positive detections and poor bounding-box localisation. Furthermore, as our method reasons globally about the image, one pixel can only be assigned to a single instance, which is not the case with FCIS. Our method also produces more precise segmentations, as it includes a differentiable CRF, and it is based off a semantic segmentation network. The results of FCIS are obtained from their publicly available results on the COCO test set (`https://github.com/daijifeng001/TA-FCN`). Note that FCIS is trained on COCO, and our model is trained on Pascal VOC which does not have as many classes as COCO, such as "umbrella" and "suitcase" among others. As a result, we are not able to detect these objects. The first column shows the input image, and the results of our object detector which are another input to our network. FCIS does not use these detections, but does internally produce proposals which are not shown. Best viewed in colour.

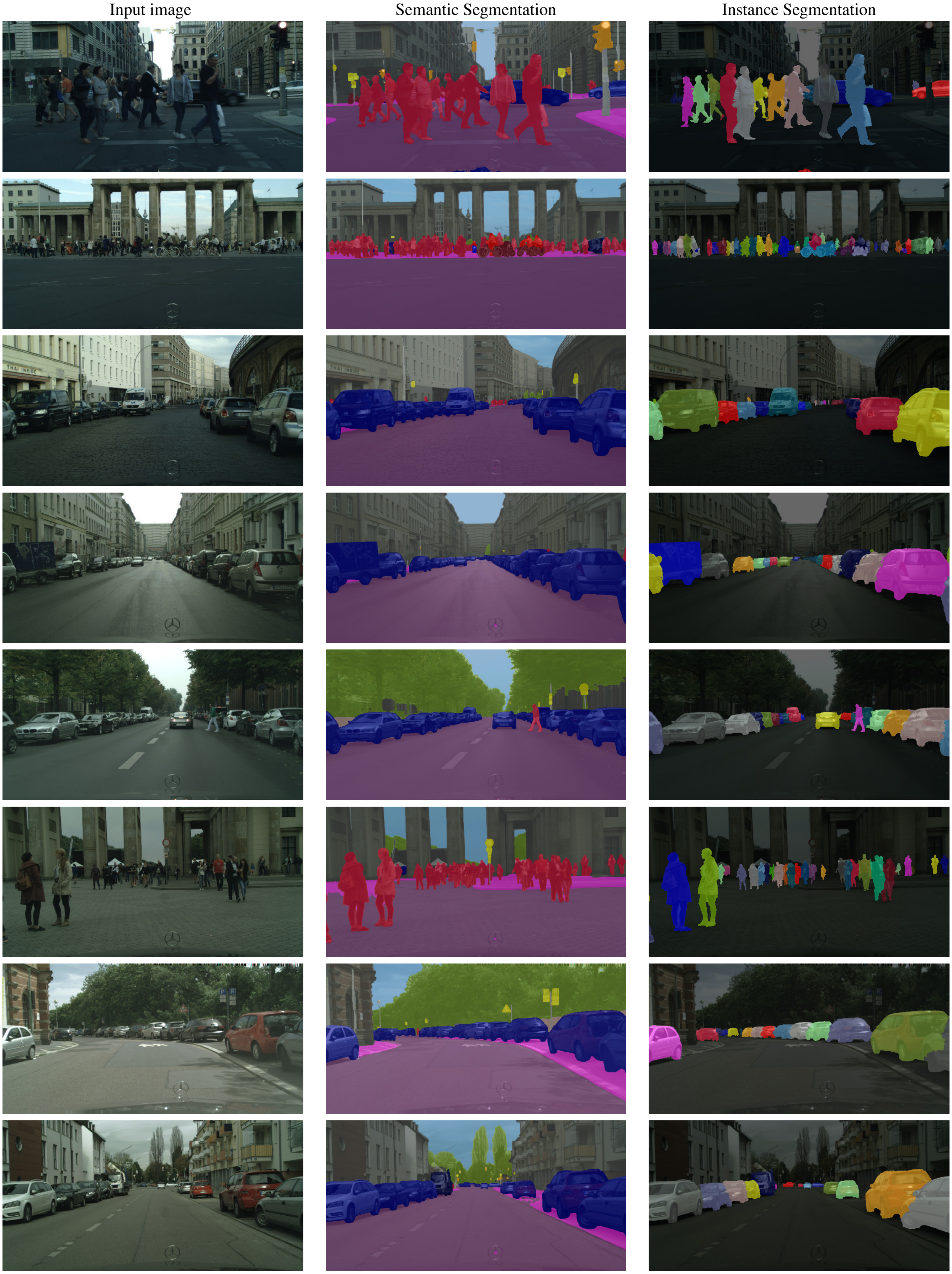

Figure 5: **Sample results on the Cityscapes dataset** The above images show how our method can handle the large numbers of instances present in the Cityscapes dataset. Unlike other recent approaches, our algorithm can deal with objects that are not continuous – such as the car in the first row which is occluded by a pole. Best viewed in colour.

2. Detailed results on the VOC dataset

Figure 6 shows a visualisation of the AP^r obtained by our method for each class across nine different thresholds. Each "column" of Fig. 6 corresponds to the AP^r for each class at a given IoU threshold. It is therefore an alternate representation for the results tables (Tables 2 to 4). We can see that our method struggles with classes such as "bicycle", "chair", "dining table" and "potted plant". This may be explained by the fact that current semantic segmentation systems (including ours) struggle with these classes. All recent methods on the Pascal VOC leaderboard [1] obtain an IoU for these classes which is lower than the mean IoU for all classes. In fact the semantic segmentation IoU for the "chair" class is less than half of the mean IoU for all the classes for 16 out of the 20 most recent submissions on the VOC leaderboard at the time of writing.

Tables 2 to 4 show per-class instance segmentation results on the VOC dataset, at IoU thresholds of 0.9, 0.7 and 0.5 respectively. At an IoU threshold of 0.9, our method achieves the highest AP^r for 16 of the 20 object classes. At the threshold of 0.7, we achieve the highest AP^r in 15 classes. Finally, at an IoU threshold of 0.5, our method, MPA 3-scale [9] and PFN [7] each achieve the highest AP^r for 6 categories.

3. Detailed results on the SBD dataset

Once again, we show a visualisation of the AP^r obtained by our method for each class across nine different thresholds (Fig. 7). The trend is quite similar to the VOC dataset in that our algorithm struggles on the same object classes ("chair", "dining table", "potted plant", "bottle"). Note that our AP^r for the "bicycle" class has improved compared to the VOC dataset. This is probably because the VOC dataset has more detailed annotations. In the VOC dataset, each spoke of a bicycle's wheel is often labelled, whilst in SBD, the entire wheel is labelled as a single circle with the "bicycle" label. Therefore, the SBD dataset's coarser labelling makes it easier for an algorithm to perform well on objects with fine details.

Table 1 shows our mean AP^r over all classes at thresholds ranging from 0.5 to 0.9. Our AP^r at 0.9 is low compared to the result which we obtained on the VOC dataset. This could be for a number of reasons: As the SBD dataset is not as finely annotated as the VOC dataset, it might not be suited for measuring the AP^r at such high thresholds. Additionally, the training data is not as good for training our system which includes a CRF and is therefore able to delineate sharp boundaries. Finally, as the SBD dataset has 5732 validation images (compared to the 1449 in VOC), it leaves less data for pretraining our initial semantic segmentation module. This may hinder our network in being able to produce precise segmentations.

Table 1: Comparison of Instance Segmentation performance at multiple AP^r thesholds on the VOC 2012 Validation Set

Method	AP^r					AP^r_{vol}
	0.5	0.6	0.7	0.8	0.9	
Ours (piecewise)	59.1	51.9	42.1	29.4	12.0	52.3
Ours (end-to-end)	**62.0**	**54.0**	**44.8**	**32.3**	**13.8**	**55.4**

Tables 5 and 6 show per-class instance segmentation results on the SBD dataset, at IoU thresholds of 0.7 and 0.5 respectively. We can only compare results at these two thresholds since these are the only thresholds which other work has reported.

[1] http://host.robots.ox.ac.uk:8080/leaderboard/displaylb.php?challengeid=11&compid=6

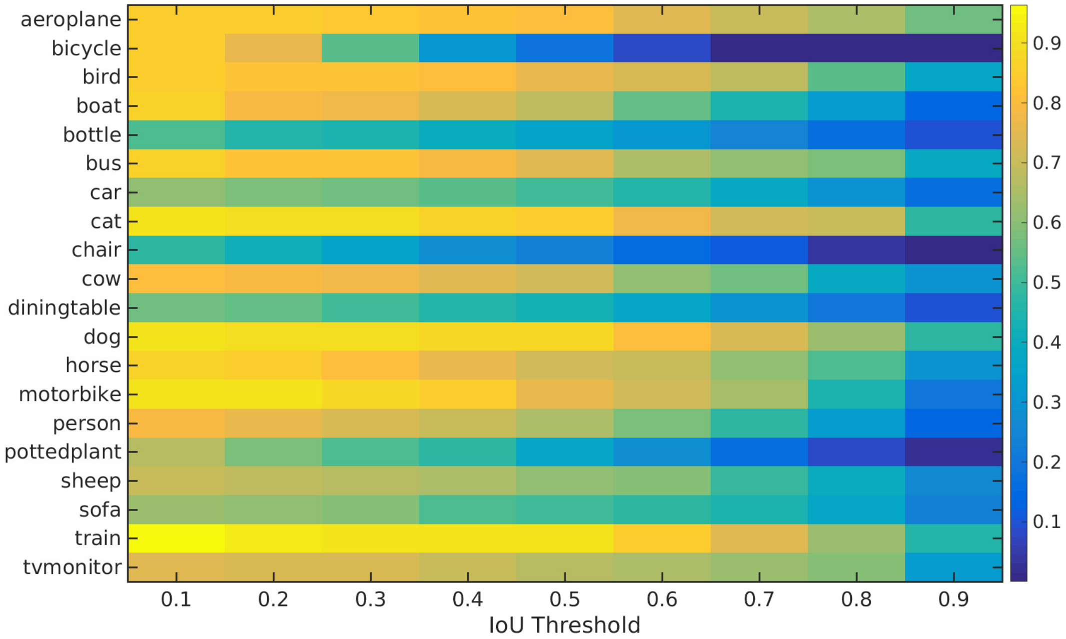

Figure 6: A visualisation of the AP^r obtained for each of the 20 classes on the VOC dataset, at nine different IoU thresholds. The x-axis represents the IoU threshold, and the y-axis each of the Pascal classes. Therefore, each "column" of this figure corresponds to the AP^r per class at a particular threshold, and is thus an alternate representation to the results tables. Best viewed in colour.

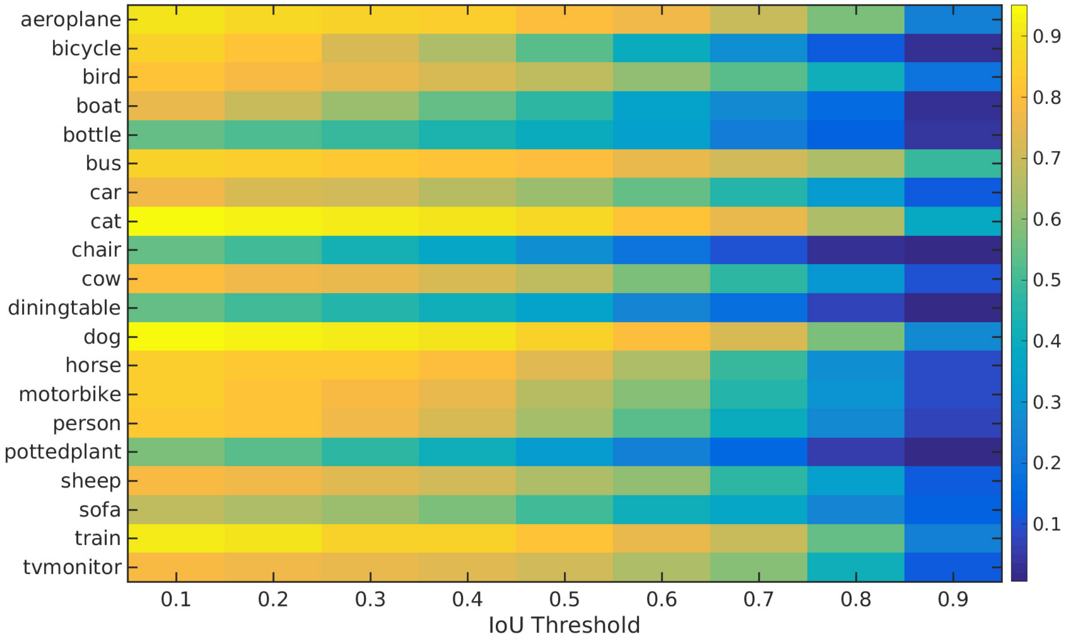

Figure 7: A visualisation of the AP^r obtained for each of the 20 classes on the SBD dataset, at nine different IoU thresholds. The x-axis represents the IoU threshold, and the y-axis each of the Pascal classes. Therefore, each "column" of this figure corresponds to the AP^r per class at a particular threshold, and is thus an alternate representation to the results tables. Best viewed in colour.

Table 2: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.9**, for all twenty classes in the VOC dataset.

Method	Mean AP^r(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	table	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	**25.1**	**56.6**	0.03	**36.8**	**14.4**	9.9	39.0	**17.2**	**47.1**	**1.3**	**29.0**	9.5	**47.2**	**29.8**	**20.0**	**14.8**	**2.3**	**25.9**	**23.8**	**45.7**	**32.3**
MPA 3-scale [9]	18.5	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
MPA 1-scale [9]	17.3	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Arnab *et al.* [1]	20.1	43.7	0.03	30.0	13.2	**11.4**	**47.3**	10.9	34.5	0.7	19.6	**12.1**	35.6	24.3	13.3	10.7	0.4	20.7	20.9	35.0	17.4
PFN [7]	15.7	43.9	**0.1**	24.5	7.8	4.1	32.5	6.3	42.0	0.6	25.7	3.2	31.8	13.4	8.1	5.9	1.6	14.8	14.3	25.0	8.5
Chen *et al.* [2]	2.6	0.6	0	0.6	0.5	4.9	9.8	1.1	8.3	0.1	1.1	1.2	1.7	0.3	0.8	0.6	0.3	0.8	7.6	4.3	6.2
SDS [4]	0.9	0	0	0.2	0.3	2.0	3.8	0.2	0.9	0.1	0.2	1.5	0	0	0	0.1	0.1	0	2.3	0.2	5.8

Table 3: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.7**, for all twenty classes in the VOC dataset.

Method	Mean AP^r(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	table	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	**48.6**	**69.6**	1.4	**68.2**	**45.1**	25.2	61.1	**38.7**	72.1	**11.2**	**56.3**	30.0	**73.3**	**60.7**	**64.3**	**46.8**	**17.1**	**49.1**	**44.6**	**75.0**	**62.0**
MPA 3-scale [9]	47.4	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
MPA 1-scale [9]	45.9	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
Arnab *et al.* [1]	45.4	68.9	0.84	65.1	38.3	**26.3**	**64.7**	31.8	72.7	6.7	45.4	**32.9**	67.9	60.0	63.7	41.1	13.4	43.9	41.1	74.6	48.1
PFN [7]	42.5	68.5	**5.6**	60.4	34.8	14.9	61.4	19.2	**78.6**	4.2	51.1	28.2	69.6	**60.7**	60.5	26.5	9.8	35.1	43.9	71.2	45.6
Chen *et al.* [2]	27.0	40.8	0.07	40.1	16.2	19.6	56.2	26.5	46.1	2.6	25.2	16.4	36.0	22.1	20.0	22.6	7.7	27.5	19.5	47.7	46.7
SDS [4]	21.3	17.8	0	32.5	7.2	19.2	47.7	22.8	42.3	1.7	18.9	16.9	20.6	14.4	12.0	15.7	5.0	23.7	15.2	40.5	51.4

Table 4: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.5**, for all twenty classes in the VOC dataset.

Method	Mean AP^r(%)	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	table	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	61.7	80.2	**19.3**	**76.4**	**69.0**	35.3	74.5	50.8	84.5	22.8	70.9	43.3	**87.7**	71.3	76.2	65.6	37.2	**61.3**	50.3	**90.5**	67.2
MPA 3-scale [9]	**62.1**	79.7	11.5	71.6	54.6	**44.7**	**80.9**	**62.0**	85.4	**26.5**	64.5	46.6	87.6	71.7	**77.9**	**72.1**	48.8	57.4	48.8	78.9	70.8
MPA 1-scale [9]	60.3	79.2	13.4	71.6	59.0	41.5	73.8	52.3	87.3	23.3	61.2	42.5	83.1	70.0	77.0	67.6	**50.7**	56.0	45.9	80.0	70.5
Arnab *et al.* [1]	58.4	**80.4**	7.9	74.4	59.8	32.7	76.6	39.6	84.6	19.3	62.7	44.1	81.0	74.7	72.0	58.6	32.0	59.6	50.5	87.4	68.4
PFN [7]	58.7	76.4	15.6	74.2	54.1	26.3	73.8	31.4	**92.1**	17.4	**73.7**	**48.1**	82.2	**81.7**	72.0	48.4	23.7	57.7	**64.4**	88.9	**72.3**
Chen *et al.* [2]	46.3	63.6	0.3	61.5	43.9	33.8	67.3	46.9	74.4	8.6	52.3	31.3	63.5	48.8	47.9	48.3	26.3	40.1	33.5	66.7	67.8
SDS [4]	43.8	58.8	0.5	60.1	34.4	29.5	60.6	40.0	73.6	6.5	52.4	31.7	62.0	49.1	45.6	47.9	22.6	43.5	26.9	66.2	66.1

Table 5: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.7**, for all twenty classes in the SBD dataset.

Method	Mean $AP^r(\%)$	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	table	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	**44.8**	**69.0**	27.4	**52.7**	**26.4**	22.4	70.3	46.0	**74.7**	9.6	46.8	**16.9**	**71.6**	48.4	46.3	**40.3**	14.8	47.6	**36.5**	**69.7**	**58.2**
IIS sp, rescore [5]	43.3	61.9	**35.1**	44.4	**26.4**	**29.6**	**74.0**	**48.7**	66.8	**10.9**	**48.4**	13.6	64.0	**53.0**	**46.8**	33.0	**19.0**	**51.0**	23.7	62.2	53.9
IIS raw [5]	38.7	61.8	31.5	42.0	22.0	22.7	72.4	44.8	65.4	7.2	37.6	10.4	60.4	39.6	41.9	32.5	12.0	40.9	19.9	58.8	50.8

Table 6: Comparison of mean AP^r, achieved by different published methods, at an IoU threshold of **0.5**, for all twenty classes in the SBD dataset.

Method	Mean $AP^r(\%)$	aero-plane	bike	bird	boat	bot-tle	bus	car	cat	chair	cow	table	dog	horse	mbike	per-son	plant	sheep	sofa	train	tv
Our method	62.0	**80.3**	52.8	68.5	47.4	39.5	79.1	61.5	**87.0**	28.1	**68.3**	**35.5**	**86.1**	73.9	66.1	63.8	32.9	65.3	**50.4**	**81.4**	**71.4**
IIS sp, rescore [5]	**63.6**	79.2	**67.9**	**70.0**	**47.9**	**45.3**	**81.6**	**68.8**	84.1	**30.4**	65.5	31.8	83.6	**75.5**	**74.5**	**66.6**	**37.7**	**70.6**	44.7	77.7	68.7
IIS raw [5]	60.1	77.3	65.3	65.5	42.5	35.4	80.3	62.2	83.9	27.2	61.6	32.4	82.3	70.9	71.4	63.1	31.3	63.6	44.9	78.3	62.4

References

[1] A. Arnab and P. H. S. Torr. Bottom-up instance segmentation with deep higher order crfs. In *BMVC*, 2016. 10

[2] Y.-T. Chen, X. Liu, and M.-H. Yang. Multi-instance object segmentation with occlusion handling. In *CVPR*, pages 3470–3478, 2015. 10

[3] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016. 1, 4, 5

[4] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*, pages 297–312. Springer, 2014. 10

[5] K. Li, B. Hariharan, and J. Malik. Iterative Instance Segmentation. In *CVPR*, 2016. 11

[6] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. In *CVPR*, 2017. 1, 6

[7] X. Liang, Y. Wei, X. Shen, J. Yang, L. Lin, and S. Yan. Proposal-free network for instance-level object segmentation. *arXiv preprint arXiv:1509.02636*, 2015. 8, 10

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, pages 740–755. Springer, 2014. 6

[9] S. Liu, X. Qi, J. Shi, H. Zhang, and J. Jia. Multi-scale patch aggregation (mpa) for simultaneous detection and segmentation. In *CVPR*, 2016. 8, 10