A Convnet for Non-Maximum Suppression

Jan Hosang, Rodrigo Benenson, Bernt Schiele

Max-Planck Institute for Informatics

Abstract. Non-maximum suppression (NMS) is used in virtually all state-of-the-art object detection pipelines. While essential object detection ingredients such as features, classifiers, and proposal methods have been extensively researched surprisingly little work has aimed to systematically address NMS. The de-facto standard for NMS is based on greedy clustering with a fixed distance threshold, which forces to trade-off recall versus precision. We propose a convnet designed to perform NMS of a given set of detections. We report experiments on a synthetic setup, crowded pedestrian scenes, and for general person detection. Our approach overcomes the intrinsic limitations of greedy NMS, obtaining better recall and precision.

1 Introduction

The bulk of object detection pipelines are based on three steps: 1) propose a set of windows (via sliding window or object proposals), 2) score each window via a trained classifier, 3) remove overlapping detections (non-maximum suppression). DPM [8] and R-CNN [12, 11, 25] follow this approach. Both object proposals [14] and detection classifiers [28] have received enormous attention, while non-maximum suppression (NMS) has been seldom addressed. The de-facto standard for NMS consists of greedily merging the higher scoring windows with lower scoring ones if they overlap enough (e.g. intersection-over-union IoU > 0.5), which we call GreedyNMS in the following.

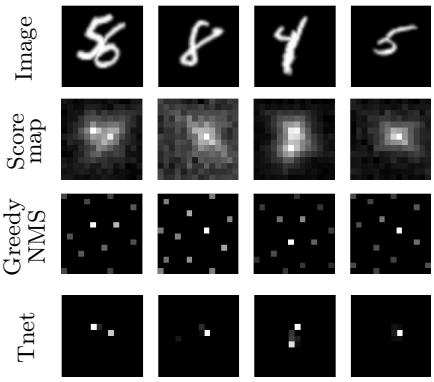


Fig. 1: GreedyNMS produces false positives and prunes true positives, while our proposed Tnet correctly localize even very close digits. First to last row: oMNIST image, input score map, GreedyNMS IoU > 0.3, and Tnet IoU & S(1, $0 \rightarrow 0.6$).

GreedyNMS is popular because it is conceptually simple, fast, and for most tasks results in satisfactory detection quality. Despite its popularity, it has important shortcomings. GreedyNMS trades off precision versus recall. If the IoU threshold is too large (too strict) then not enough surrounding detections are suppressed, high scoring false positives are introduced and precision suffers. If the IoU threshold is too low (too loose) then multiple true positives are merged

together and the recall suffers. For any IoU threshold, GreedyNMS is sacrificing precision or recall (as shown experimentally in §4). One can do better than this by leveraging the full signal of the score map (statistics of the surrounding detections) rather than blindly applying a fixed policy everywhere in the image.

Current object detectors are becoming surprisingly effective on both general (e.g. Pascal VOC, COCO) and specific object detection (e.g. pedestrians, faces). The oracle analyses for "perfect NMS" from [14, table 5] and [23, figure 12] both indicate that NMS accounts for almost a quarter of the remaining mistakes.

Instead of doing hard pruning decisions as GreedyNMS, we design our network to make soft decisions by re-scoring (re-ranking) the input detection windows. Our re-scoring is final, and no post-processing is done afterwards, thus the resulting score maps must be very "peaky". We call our proposed network "Tyrolean network", abbreviated Tnet. (Tyrolean because "it likes to see peaks".)

Contribution We are the first to show that a convnet can be trained and used to overcome the limitations of GreedyNMS. Our experiments demonstrate that, across different occlusion levels, the Tyrolean network (Tnet) performs strictly better than GreedyNMS at *any* IoU threshold.

As an interesting scenario for NMS, we report results for crowded pedestrian scenes and general person detection. Our Tnet can operate solely over detection boxes (like GreedyNMS), and does not use external training data. Furthermore, Tnet provides better results than auto-context [37]. We consider our results a proof of concept, opening the door for further exploration.

1.1 Related work

Clustering detections The decade old greedy NMS (GreedyNMS) is used in popular detectors such as V&J [39], DPM [8], and is still used in the state-of-the-art R-CNN detector family [12, 11, 25]. Alternatives such as mean-shift clustering [5, 42], agglomerative clustering [2], and heuristic variants [30] have been considered, but they have yet to show consistent gains. Recently [35, 27] proposed principled clustering approaches that provide globally optimal solutions, however the results reached are on par, but do not surpass, GreedyNMS.

Linking detections to pixels The Hough voting framework enables reasoning amongst conflicting detections by linking the detections to local image evidence [18, 1, 17, 41]. Hough voting itself, however, provides low detection accuracy. [44, 4] refine detections by linking them with semantic labelling; while [43] side-steps NMS all-together by defining the detection task directly as a labelling problem. These approaches arguably propose a sound formulation of the detection problem, however they rely on semantic labelling/image segmentation. Our system operates directly on bounding box detections.

Co-occurrence To better handle dense crowds or common object pairs, it has been proposed to use specialized 2-object detectors [29, 34, 22], which then require a careful NMS strategy to merge single-object with double-object detections. Similarly, [26] adapts the NMS threshold using crowd density estimation. Our approach is directly learning based (no hand-crafted 2-objects or density estimators), and does not use additional image information.

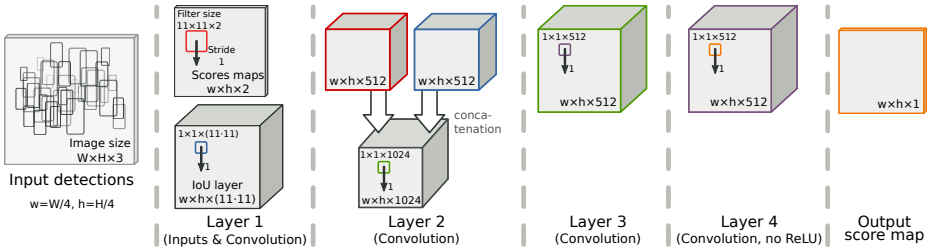


Fig. 2: Base architecture of our Tyrolean network (Tnet). Each box is a feature map, its dimensions are indicated at its bottom, the coloured square indicates the convolutional filters size, the stride is marked next to the downward arrow.

Auto-context re-score detections using local [37, 3] or global [38] image information. Albeit such approaches do improve detection quality, they still require a final NMS processing step. Our convnet does re-score detections, but at the same time outputs a score map that does not require further processing. We provide experiments (in §4) that show improved performance over auto-context.

Convnets and NMS Few works have linked convnets and NMS, detection convnets are commonly trained unaware of the NMS post-processing step. [40] proposed an NMS-aware training loss, making the training truly end-to-end. The used NMS is greedy and with fixed parameters. [32] proposes to use an LSTM to decide how many detections should be considered in a local region. The detections amongst the regions are then merged via traditional NMS. In contrast, our convnet requires no post-processing. To the best of our knowledge our Tnet is the first network explicitly designed to replace the final NMS stage.

In §2 we describe our base network, §3 explores its use in a synthetic setup. Then §4 reports results over DPM [8] detections in crowds datasets (small scale variance), and finally results over FasterRCNN [25] on Pascal VOC people [7].

2 Base Tyrolean network

The main intuition behind our proposed Tyrolean network (Tnet) is that the score map of a detector together with a map that represents the overlap between neighbouring detections contains valuable information to perform better NMS than GreedyNMS (see figure 1, second row). Our network is a traditional convnet but with access to two slightly unusual inputs (described below), namely score map information and IoU maps. Figure 2 shows the overall network. In our base Tnet the first stage applies 512 11×11 filters over each input layer, and 512 1×1 filters are applied on layers 2 and 3. ReLU non-linearities are used after each layer but the last one. Neither max-pooling nor local normalization is used.

The base network is trained and tested in a fully convolutional fashion. It uses the same information as GreedyNMS, and does not access the image pixels directly. The required training data are only a set of object detections (before

NMS), and the ground truth bounding boxes of the dataset. We focus on the single class case and consider exploiting multi-class information future work.

Input grid As preprocessing all detections in an image are mapped into a 2d grid (based on their centre location). If more than one detection falls into the same cell, we keep only the highest scoring detection. Each cell in the grid is associated with a detection bounding box and score. We use cells of 4×4 pixels, thus an input image of size $\mathrm{W} \times \mathrm{H}$ will be mapped to input layers of size $\mathrm{w} \times \mathrm{h} = {}^{\mathrm{W}}/_{4} \times {}^{\mathrm{H}}/_{4}$. Since the cells are small, mapping detections to the input grid has minimal impact on the NMS procedure. In preliminary experiments we validated that: a) we can at least recover the performance of GreedyNMS (applying GreedyNMS over the input grid provides the same results as directly applying GreedyNMS), b) the detection recall stays the same (after mapping to the input grid the overall recall is essentially identical to the raw detections).
This incarnation of Tnet can handle mild changes in scale amongst neighbouring detections. §4 reports experiments with detections over a $3\times$ scale range. In §4 we also explain how to adapt our approach to general person detection (Pascal VOC [7]), with large scale and aspect ratio variance.

IoU layer In order to reason about neighbouring detection boxes (or segments) we feed Tnet with IoU values. For each location we consider a $11 \times 11 = 121$ neighbourhood, thus the input IoU layer has $\mathrm{w} \times \mathrm{h} \times 121$ values. Together the cell size and neighbourhood size should provide the Tnet with sufficient information about surroundings of a detection, where this choice depends on the object sizes in the image and the expected object density and thus are application dependent.

Score maps layer To reason about the detection confidence, we feed Tnet with the raw detection score map (once mapped to the input grid). The NMS task involves ranking operations which are not easily computed by linear and ReLU ($\max(\cdot, 0)$) operators. To ease the task we also feed the Tnet with score maps resulting from GreedyNMS at multiple IoU thresholds. All score maps are stacked as a multi-channel input image and feed into the network. $\mathrm{S}(\tau)$ denotes a score map resulting from applying GreedyNMS with IoU$\geq \tau$, $\mathrm{S}(\tau_1, \tau_2)$ denotes a two channels map ($\mathrm{S}(\tau_1)$ and $\mathrm{S}(\tau_2)$ stacked). Note that $\mathrm{S}(1)$ returns the raw detection score map. Our base Tnet uses $\mathrm{S}(1, 0.3)$ which has dimensionality $\mathrm{w} \times \mathrm{h} \times 2$ (see figure 2). The convolutional filters applied over the score maps input have the same size as the IoU layer neighbourhood (11×11 cells).

Tnet is then responsible for interpreting the multiple score maps and the IoU layer, and make the best local decision. Our Tnet operates in a fully feed-forward convolutional manner. Each location is visited only once, and the decision is final. In other words, for each location the Tnet has to decide if a particular detection score corresponds to a correct detection or will be suppressed by a neighbouring detection in a single feed-forward path.

Parameter rules of thumb Figure 2 indicates the base parameters used. Preliminary experiments indicated that removing top layers has a clear negative impact on the network performance, while the width of these layers is rather insensitive. Having a high enough resolution in the input grid is critical, while keeping

a small enough number of convolutions over the inputs allows to keep the number of model parameters under control. During training data augmentation is necessary to avoid overfitting. The training procedure is discussed in §2.1, while experimental results for some parameters variants are reported in §4.

Input variants Experiments in the next sections consider multiple input variants. The IoU layer values can be computed over bounding boxes (regressed by the sliding window detector) or over estimated instance segments [24]. Similarly, for the score maps we consider different numbers of GreedyNMS thresholds, which changes the dimensionality of the input score map layer.
In all cases we expect the Tnet to improve over a fixed threshold GreedyNMS by discovering patterns in the detector score maps and IoU arrangements that enable to do adaptive NMS decisions.

2.1 Training procedure

Typically detectors are trained as classifiers on a set of positive and negative windows, determined by the IoU between detection and object annotation. When doing so the spatial relation between detector outputs and the annotations is neglected. We adopt the idea from [32] of computing the loss by matching detections to annotations, and train the network to predict new detection scores that are high for matched detections and low everywhere else. In contrast to the conventional wisdom of training the detector to have a smooth score decrease around positive instances, we declare a detection right next to a true positive to be a negative training sample. Processing detections *independently* would hurt generalisation, but Tnet has access to neighbouring detections circumventing this problem. This is necessary because our network must itself perform NMS.

Training loss Our goal is to reduce the score of all detections that belong to the same person, except exactly one of them. To that end, we match every annotation to the highest scoring detection that overlaps at least 0.5 IoU. This determines the set of positives, while all other detections are negative training examples. This yields a label y_p for every location p in the input grid (see previous section). Since background detections are much more frequent than true positives, it is necessary to weight the loss terms to balance the two. We use the weighted logistic loss and choose the weights so that both classes have the same weight per frame. We also consider setting weights to balance classes across the full dataset and giving lower weights for highly occluded samples, see §4.1.

The model is trained from scratch, randomly initialized with MSRA [13], and optimized via Adam [16]. All experiments are implemented with Caffe [15]. See supplementary material for details of the training loss and training parameters.

As pointed out in [20] the threshold for GreedyNMS requires to be carefully selected on the validation set of each task, the commonly used default IoU > 0.5 can severely underperform. Other NMS approaches such as [35, 27] also require training data to be adjusted. When maximizing performance in cluttered scenes is important, training a Tnet is thus not a particularly heavy burden. Training our base Tnet on un-optimized CPU and GPU code takes a day.

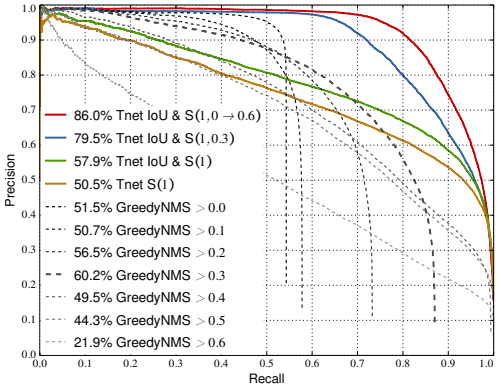


Fig. 3: oMNIST test set detection results.

Method	AR
GreedyNMS	
bboxes IoU > 0.3	54.3%
DeepMask segments	52.0%
Tnet variants	
IoU & S(1, $0 \rightarrow 0.6$)	*59.6%*
IoU & S(1, 0.3)	57.9%
IoU & S(1)	36.5%
S(1)	33.9%

Table 1: PETS val. set results. Base Tnet is underlined.

3 Controlled setup experiments

NMS is usually the last stage of an entire detection pipeline. Therefore, in an initial set of experiments, we want to understand the problem independent of a specific detector and abstract away the particularities of a given dataset.

If objects appeared alone in the images, NMS would be trivial. The core issue for NMS is deciding if two local maxima in the detection score map correspond to one or multiple objects. To investigate this core aspect we create the oMNIST ("overlapping MNIST") toy dataset. This data does not aim at being particularly realistic, but rather to enable a detailed analysis of the NMS problem.
Each image is composed of one or two off-centre MNIST digits with IoU $\in [0.2, 0.6]$. We mimic a detector by generating synthetic perturbed score maps. Albeit noisy, the detector is "ideal" because its detection score remains high despite strong occlusions. The supplementary material and figure 1 show examples of the generated score maps and corresponding images. By design GreedyNMS will have difficulties handling such cases (at any IoU threshold). We generate a training/test split of 100k/10k images (fix across experiments).

Other than score maps our convnet uses IoU information between neighbouring detections (like GreedyNMS). Our experiments cover using the perfect segmentation masks for IoU (ideal case), noisy segmentation masks, and the sliding window bounding boxes.

3.1 Results

Results are summarised in figure 3. Curves are scored via AR; the average recall on the precision range [0.5, 1.0]. The evaluation is done using the standard Pascal VOC protocol, with IoU > 0.5 [7].

GreedyNMS As can be seen in figure 3 varying the IoU thresholds for GreedyNMS trades off precision and recall. The best AR that can be obtained with GreedyNMS is 60.2% for IoU > 0.3. Example score maps for this method can be found in figure 1, third row.

Upper bound As an upper bound for any method relying on score map information we calculate the overlap between neighbouring hypotheses based on perfect segmentation masks (available in this toy scenario). With perfect overlaps and perfect scores GreedyNMS returns perfect results. Based on our idealized but noisy detection score maps the upper bound reaches 90.0% AR. In §4 we report experiments using segmentation masks estimated from the image, which results in inferior performance.

Base Tnet Using the same information as GreedyNMS with bounding boxes, our base Tnet reaches better performance for the entire recall range (see figure 3), S(1, 0.3) indicates the score maps from GreedyNMS with IoU > 0.3 and ≥ 1, i.e. the raw score map. In this configuration Tnet obtains 79.5% AR, clearly superior to GreedyNMS. This shows that, at least in a controlled setup, a convnet can indeed exploit the available information to overcome the limitations of the popular GreedyNMS method.

Instead of picking a specific IoU threshold to feed Tnet, we consider IoU & S(1, $0 \rightarrow 0.6$), which includes S(1, 0.6, 0.4, 0.3, 0.2, 0.0). As seen in figure 3, not selecting a specific threshold results in the best performance; 86.0% AR. If we remove GreedyNMS score maps and only provide the raw score map (IoU & S(1)) performance decreases significantly. As soon as some ranking signal is provided (via GreedyNMS score maps), our Tnet is able to learn how to exploit best the information available. Qualitative results are presented in figure 1, bottom row.

Auto-context Importantly we show that IoU & S(1) improves over S(1) only. (S(1) is the information exploited by auto-context methods, see §1.1). This shows that the convnet is learning to do more than simple auto-context. The detection improves not only by noticing patterns on the score map, but also on how the detection boxes overlap.

4 Person detection experiments

After the proof of concept in a controlled setup, we move to a realistic pedestrian detection setup. We are particularly interested in datasets that show diverse occlusion where NMS is non-trivial. We decided for the PETS dataset [9], which exhibits diverse levels of occlusion and provides a reasonable volume of training and test data. We use 5 sequences for training, one sequence for validation and testing (23k, 4k, 10k annotations respectively, see supplementary material for details). PETS has been previously used to study person detection [36], tracking [21], and crowd density estimation [33]. Additionally we test the generalization of the trained model on the ParkingLot dataset [31], and the applicability to general person detections on Pascal VOC [7]. Figure 6 shows example frames.

Standard pedestrian datasets such as Caltech [6] or KITTI [10] average less than two pedestrians per frame, making close-by detections a rare occurrence. In PETS and ParkingLot $> 50\%$ of pedestrians have some occlusion, and about $\sim 20\%$ have significant occlusion (IoU > 0.4). Pascal presents fewer occlusion cases, people being the class where it is most frequent. See supplementary material for details on these datasets.

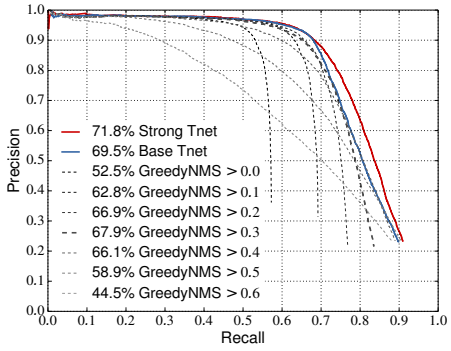


Fig. 4: Detection results on PETS test set. Our approach is better than any GreedyNMS threshold and better than the upper envelope of all GreedyNMS curves.

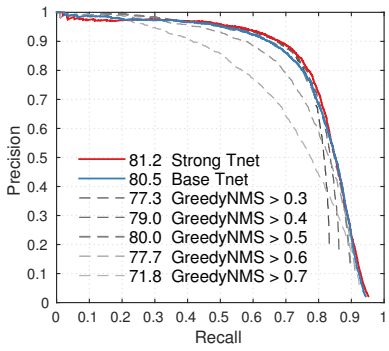


Fig. 5: Persons detection results on Pascal'12 test set. Tnet improves over all GreedyNMS thresholds.

Person detector In this work we take the detector as a given. For the PETS experiments we use the baseline DPM detector from [36]. We are not aware of a detector (convnet or not) providing better results on PETS-like sequences (we considered some of the top detectors in [6]). Importantly, for our exploration the detector quality is not important per-se. As discussed in §3 GreedyNMS suffers from intrinsic issues, even when providing an idealized detector. In fact Tnet benefits from better detectors, since there will be more signal in the score maps. We thus consider our DPM detector a fair detection input. We use the DPM detections after bounding box regression, but before any NMS processing.

Person segments In §4.1 we report results using segments estimated from the image content. We use our re-implementation of DeepMask [24], trained on the Coco dataset [19]. See supplementary material for details and qualitative results. We use DeepMask as a realistic example of what can be expected from modern techniques for instance segmentation.

4.1 Results

Our PETS results are presented in table 1 (validation set) and figure 4 (test set). Qualitative results are shown in figure 6. The supplementary material provides additional val & test results.

Boxes Just like in the oMNIST case, the GreedyNMS curves in figure 4 have a recall versus precision trade-off. We pick IoU >0.3 as a reference threshold.

Segments GreedyNMS should behave best when the detection overlap is based on the visible area of the object. We compute DeepMask segments over DPM detection, feed these in GreedyNMS, and select the best IoU threshold for the validation set. Table 1 shows results slightly below the bounding boxes case. Although many segments are rather accurate, they drop in quality when heavier occlusion is present. In theory using segments should improve GreedyNMS, in practice they hurt more than they help.

Auto-context For the S (1) entry in table 1 only the raw detection score map is feed to Tnet (same nomenclature as §3.1). Since performance is lower than other variants (e.g. IoU & S (1)), this shows that our approach is exploiting available information better than just doing auto-context over DPM detections.

Tnet Both in validation and test set our trained network with IoU & S (1, 0.3) input provides a clear improvement over vanilla GreedyNMS. Just like in the oMNIST case, the network is able to leverage patterns in the detector output to do better NMS than the de-facto standard GreedyNMS method.
Table 1 reports the results for a few additional variants. IoU & S(1, $0 \rightarrow 0.6$) shows that it is not necessary to select a specific IoU threshold for the input score map layer. Given an assortment (S(1, 0.6, 0.4, 0.3, 0.2, 0.0)) the network will learn to leverage the information available.
Using a relaxed loss that decreases weight of hard examples (peaks on the background and strong occlusions) helps further improve the results, moving from 57.9% to 58.9% AR. Weighting classes equally over the full dataset (global weighting) instead of frame-by-frame gives a mild improvement from 57.9% to 58.0% AR. See supplementary material for details on these loss variants.

Strong Tnet We combine the best ingredients identified on the validation set into one strong model. We use IoU & S(1, $0 \rightarrow 0.6$), relaxed loss, and global weighting. Figure 4 shows that we further improve over the base Tnet from 59.5% to 71.8% AR on the PETS test set. The gap between base Tnet and GreedyNMS is smaller on the test set than on validation, because test set has lighter occlusions. Still our strong Tnet provides a consistent improvement over GreedyNMS.
In the supp. material we experimentally show that Strong Tnet improves GreedyNMS for all occlusions levels. Our network does not fit to a particular range of occlusions, but learns to handle all of them with comparable effectiveness.
At test time Tnet takes $\sim$200 milliseconds per frame (all included).

ParkingLot results To verify that our Tnet can generalize beyond PETS, we run the same DPM detector as on the PETS experiment over the ParkingLot sequence and do NMS using the networks trained on PETS training set only. Results show that Tnet improves from 80.3% to 83.3% AR over the best GreedyNMS threshold of IoU > 0.3. Even though Tnet was not trained on this sequence we see a similar result as on the PETS dataset. Not only does our Strong Tnet improve over the best GreedyNMS result, but it improves over the upper envelope of all GreedyNMS thresholds (similar trend as figure 4). See see detailed curves in supplementary material and qualitative results in figure 6.

Pascal results Pascal VOC [7] contains less occlusion but is more challenging with respect to appearance, scale, and aspect ratio variance. We focus on the "people" class which offers the highest diversity in occlusion. As a base detector we use the publicly available FasterRCNN [25]. The small variance in performance of the GreedyNMS swipe in figure 5 shows that this data contains fewer occlusions than PETS.
Tnet is trained on Pascal '07 trainval and tested on the test set. To adapt the Tnet to multiple scales and aspect ratio we switch from the fully convolutional approach to a detection-centric representation. Instead of a fixed-size

neighbourhood grid we adapt its scale and aspect ratio to each detection box being re-scored. We also use the image features. See details in the supplementary material. After tuning the training parameters, Base Tnet matches the best GreedyNMS with 80.5% AP (figure 5). Strong Tnet matches the upper envelope of all GreedyNMS thresholds, improving the results to 81.2% AP.

5 Conclusion

We have discussed the limitations of GreedyNMS in detail and presented experiments showing its recall versus precision trade-off. For the sake of speed and simplicity GreedyNMS disregards most of the information available in the detector response. Our proposed Tyrolean network (Tnet) mines the patterns in the score map values and bounding box arrangements to surpass the performance of GreedyNMS. On the person detection task, our final results show that our approach provides, compared to any GreedyNMS threshold, both high recall and improved precision. These results confirm that Tnet can overcome the intrinsic limitations of GreedyNMS, while keeping practical test time speeds. We consider the reported results a proof of concept, opening the door for further extensions.

Current detection pipelines consist of a convnet and a hard-coded NMS procedure. Replacing the NMS with a Tnet opens the possibility of true end-to-end training of object detectors and we reckon that significant improvements can be obtained by replacing NMS with a Tnet.

(a) GreedyNMS (b) Strong Tnet

Fig. 6: Qualitative detection results of GreedyNMS and Strong Tnet (both operating at same recall). Tnet is able to suppress false positives as well as recover recall that is lost with GreedyNMS. See supp. material for additional results.

References

1. Barinova, O., Lempitsky, V., Kholi, P.: On detection of multiple object instances using hough transforms. PAMI (2012)
2. Bourdev, L., Maji, S., Brox, T., Malik, J.: Detecting people using mutually consistent poselet activations. In: ECCV (2010)
3. Chen, G., Ding, Y., Xiao, J., Han, T.X.: Detection evolution with multi-order contextual co-occurrence. In: CVPR (2013)
4. Dai, J., He, K., Sun, J.: Convolutional feature masking for joint object and stuff segmentation. In: CVPR (2015)
5. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR (2005)
6. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. PAMI (2012)
7. Everingham, M., Eslami, S.M.A., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. IJCV (2015)
8. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. PAMI (2010)
9. Ferryman, J., Ellis, A.: Pets2010: Dataset and challenge. In: AVSS (2010)
10. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR (2012)
11. Girshick, R.: Fast R-CNN. In: ICCV (2015)
12. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
13. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: ICCV (2015)
14. Hosang, J., Benenson, R., Dollár, P., Schiele, B.: What makes for effective detection proposals? PAMI (2015)
15. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: ACM International Conference on Multimedia (2014)
16. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
17. Kontschieder, P., Rota Bulò, S., Donoser, M., Pelillo, M., Bischof, H.: Evolutionary hough games for coherent object detection. CVIU (2012)
18. Leibe, B., Leonardis, A., Schiele, B.: Robust object detection with interleaved categorization and segmentation. IJCV (2008)
19. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
20. Mathias, M., Benenson, R., Pedersoli, M., Van Gool, L.: Face detection without bells and whistles. In: ECCV (2014)
21. Milan, A., Roth, S., Schindler, K.: Continuous energy minimization for multitarget tracking. PAMI (2014)
22. Ouyang, W., Wang, X.: Single-pedestrian detection aided by multi-pedestrian detection. In: CVPR (2013)
23. Parikh, D., Zitnick, C.: Human-debugging of machines. In: NIPS WCSSWC (2011)
24. Pinheiro, P.O., Collobert, R., Dollar, P.: Learning to segment object candidates. In: NIPS (2015)
25. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS (2015)

26. Rodriguez, M., Laptev, I., Sivic, J., Audibert, J.Y.: Density-aware person detection and tracking in crowds. In: ICCV (2011)
27. Rothe, R., Guillaumin, M., Van Gool, L.: Non-maximum suppression for object detection by passing messages between windows. In: ACCV (2015)
28. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet Large Scale Visual Recognition Challenge. IJCV (2015)
29. Sadeghi, M.A., Farhadi, A.: Recognition using visual phrases. In: CVPR (2011)
30. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., LeCun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks. In: ICLR (2014)
31. Shu, G., Dehghan, A., Oreifej, O., Hand, E., Shah, M.: Part-based multiple-person tracking with partial occlusion handling. In: CVPR (2012)
32. Stewart, R., Andriluka, M.: End-to-end people detection in crowded scenes. arXiv:1506.04878 (2015)
33. Subburaman, V.B., Descamps, A., Carincotte, C.: Counting people in the crowd using a generic head detector. In: AVSS (2012)
34. Tang, S., Andriluka, M., Schiele, B.: Detection and tracking of occluded people. In: BMVC (2012)
35. Tang, S., Andres, B., Andriluka, M., Schiele, B.: Subgraph decomposition for multi-target tracking. In: CVPR (2015)
36. Tang, S., Andriluka, M., Milan, A., Schindler, K., Roth, S., Schiele, B.: Learning people detectors for tracking in crowded scenes. In: ICCV (2013)
37. Tu, Z., Bai, X.: Auto-context and its application to high-level vision tasks and 3d brain image segmentation. PAMI (2010)
38. Vezhnevets, A., Ferrari, V.: Object localization in imagenet by looking out of the window. In: BMVC (2015)
39. Viola, P., Jones, M.: Robust real-time face detection. In: IJCV (2004)
40. Wan, L., Eigen, D., Fergus, R.: End-to-end integration of a convolutional network, deformable parts model and non-maximum suppression. In: CVPR (2015)
41. Wohlhart, P., Donoser, M., Roth, P.M., Bischof, H.: Detecting partially occluded objects with an implicit shape model random field. In: ACCV (2012)
42. Wojek, C., Dorkó, G., Schulz, A., Schiele, B.: Sliding-windows for rapid object class localization: A parallel technique. In: DAGM (2008)
43. Yan, J., Yu, Y., Zhu, X., Lei, Z., Li, S.Z.: Object detection by labeling superpixels. In: CVPR (2015)
44. Yao, J., Fidler, S., Urtasun, R.: Describing the scene as a whole: Joint object detection, scene classification and semantic segmentation. In: CVPR (2012)

A Convnet for Non-Maximum Suppression Supplementary Material

Jan Hosang, Rodrigo Benenson, Bernt Schiele

Max-Planck Institute for Informatics

1 Content

This supplementary material provides expanded explanations and describes implementation details, it provides as well additional quantitative and qualitative results. In particular:

- section 2 discusses the limitation of GreedyNMS, and why it trades-off precision versus recall,
- section 3 details the training loss and parameters,
- section 4 describes the used datasets,
- section 5 explains in more detail how we modify the Tnet architecture to handle larger scale and aspect ratio variance,
- section 6 details our DeepMask re-implementation,
- section 7 provides additional result tables and curves from our main experiments,
- and section 8 presents corresponding qualitative results.

2 Precision versus recall GreedyNMS threshold

This section contains a more in-depth explanation of the fundamental limitation of GreedyNMS that did not find space in the main paper.

Consider the example figure 1. A narrow suppression correctly returns detections for the two objects, but also introduces a high scoring false positive that actually has a higher score than the lower scoring detection. Making the suppression threshold wider decreases the score of the false positive, but eventually also removes the correct, lower scoring detection. In this example there is no one threshold that resolves this problem and GreedyNMS is doomed to fail.

Note that this situation does not only arise inside a single image. The problem of a high scoring false positive surviving suppression can happen on a different image than the object that is wrongly suppressed due to high overlap. In general, whenever the merge criterion is too narrow false positives will be introduced (low precision, middle case of figure 1), when the merge criterion is too wide there are missed detections (low recall, right case). No one threshold will be suitable across different occlusion levels.

The main paper, as well as figures 6, 7, and 9 confirm experimentally the precision versus recall trade-off of GreedyNMS.

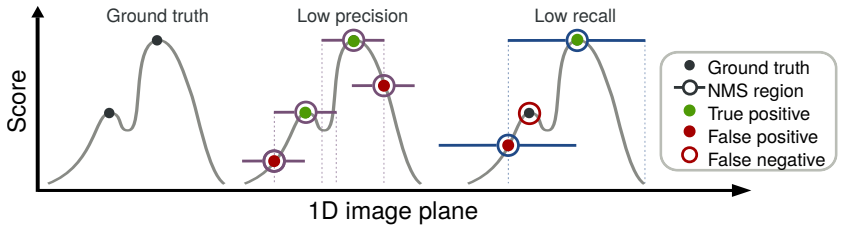


Fig. 1: 1D illustration of the GreedyNMS shortcomings. Black dots indicate true objects, grey curve is the detector response, green dots are true positives, red dots/circles are false positives/negatives.

3 Training procedure details

Training loss Our goal is to reduce the score of all detections that belong to the same person, except exactly one of them. To that end, we match every annotation the to highest scoring detection that overlaps at least 0.5 IoU. This yields a label y_p for every location p in the input grid (see section 2, input grid). Since background detections are much more frequent than true positives, it is necessary to weight the loss terms to balance the two. We use the weighted logistic loss

$$L(\mathbf{x}) = \sum_{p \in G} w_{y_p} \log\left(1 + e^{-y_p f(x_p)}\right) \tag{1}$$

where x_p is the feature descriptor at position p and $f(x_p)$ is the prediction of the network at position p. The weights w_{y_p} are chosen so that both classes have the same weight either per frame or globally on the entire dataset (denoted by w_f and w_g respectively). Since we have a one-to-one correspondence between input grid cells and labels it is straight forward to train a fully convolutional network to minimize this loss.

Relaxed loss It is impossible for the network to recover from certain mistakes that are already present in the input detections. For example, false positives on the background might be impossible to tell apart from true positives since the network does not have access to the image and only sees detection scores and overlaps between detections. On the other hand detections of distinct objects with high overlap can be hard to detect since the detections can assign low scores to barely visible objects. It proved beneficial to assign lower weight to these cases, which we call the *relaxed loss*. We declare negative samples to be hard if the corresponding detections are not suppressed by a 0.3 NMS and true positives to be hard if they are suppressed by a 0.3 NMS on the annotations with the matched detection scores. The weight of hard examples is decreased by a factor of r. Our base Tnet uses $r = 1$ (non-relaxed) with weighting strategy w_f, and section 7.2 reports results for other r values and w_g.

3.1 Training parameters

PETS The model is trained from scratch, randomly initialized with MSRA [6], and optimized with Adam [8]. We use a learning rate of 10^{-4}, a weight decay of $5 \cdot 10^{-5}$, a momentum of 0.9, and gradient clipping at 1 000. The model is trained for 100 000 iterations with one image per iteration. All experiments are implemented with the Caffe framework [7].

Since detectors tend to have non-zero detection scores in most areas of the image, the training volume is proportional to number of pixels not the number of images. Thus we can adequately train our Tnet with only a few hundred frames.

Pascal VOC For Pascal, since we are using a different architecture, parameters are slightly different. We weight decay of $5 \cdot 10^{-3}$ and for 10 000 iterations with two images per iteration.

4 Dataset details

Figure 2 shows the overlap distribution of the different datasets considered in this paper. Most annotations on ParkingLot and in particular on PETS have some small amount of occlusion. People in Pascal VOC '07 have some occlusion in half of the cases, in Caltech only 20% of the pedestrians. For PETS and ParkingLot, about 20% of the test set have significant occlusion (IoU > 0.4). On Pascal only about 5% of the test set has significant occlusion, on Caltech even less. Due to these statistics we focus our analysis on PETS.

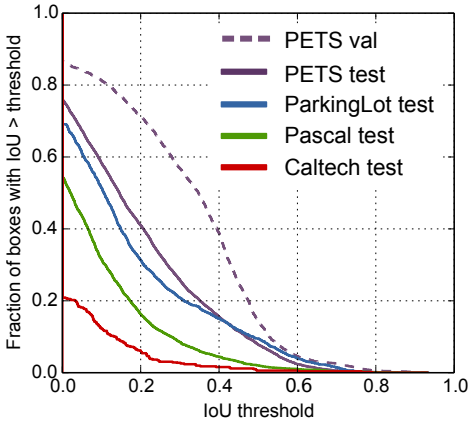


Fig. 2: Distribution of IoU overlap between ground truth annotations, on the datasets discussed in the paper.

4.1 oMnist

If all objects appeared alone in the images, NMS would be trivial. The core issue for NMS is deciding if two local maxima in the detection score map correspond to only one object or to multiple ones. To investigate this core aspect we create the oMNIST ("overlapping MNIST") toy dataset. This data does not aim at being particularly realistic, but rather to enable a detailed analysis of the NMS problem.

Each image is composed of one or two MNIST digits. To emphasise the occlusion cases, we sample $1/5$ single digits, and $4/5$ double digit cases. The digits are off-centre and when two digits are present they overlap with bounding box IoU $\in [0.2, 0.6]$. We also mimic a detector by generating synthetic score maps. Each ground truth digit location generates a perturbed bump with random magnitude in the range $[1, 9]$, random x-y scaling, rotation, a small translation, and additive Gaussian noise. Albeit noisy, the detector is "ideal" since its detection

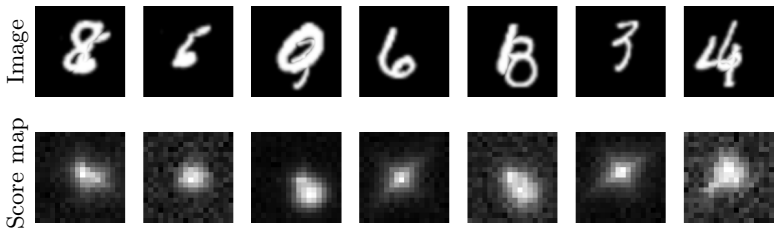


Fig. 3: Example data from our controlled experiments setup. The convnet must decide if one or two digits are present (and predict is their exact location) while using only a local view of score and IoU maps (no access to the input image).

score remains high despite strong occlusions. Figure 3 shows examples of the generated score maps and corresponding images. By design GreedyNMS will have difficulties handling such cases (at any IoU threshold).

Other than score maps our convnet uses IoU information between neighbouring detections (like GreedyNMS). In our experiments we consider using the perfect segmentation masks for IoU (ideal case), noisy segmentation masks, and the sliding window bounding boxes.

We generate a training/test split of 100k/10k images, kept fix amongst different experiments.

4.2 PETS

We use 8 of the PETS sequences [3], $\sim$200 frames each, that we split in 5 for training (S1L1-1, S1L1-2, S1L2-1, S1L2-2, S2L1, and S3MF1), 1 for validation (S2L3) and 1 for testing (S2L2). The different videos show diverse densities of crowds. As shown in figure 2 more than 40/50/25% of the train/val/test data has an IoU > 0.3 with another ground truth box.
Since detectors tend to have non-zero detection scores in most areas of the image, the training volume is proportional to number of pixels not the number of images. Thus we can adequately train our Tnet with only a few hundred frames.
PETS has been previously used to study person detection [14], tracking [10], and crowd density estimation [13]. Standard pedestrian datasets such as Caltech [1] or KITTI [4] average less than two pedestrian per frame, making close-by detections a rare occurrence.

Due to its size and challenging occlusion statistics we consider PETS a suitable dataset to explore NMS. Figure 10 shows example frames.

4.3 ParkingLot

We use the first ParkingLot [12] sequence to evaluate the generalization capabilities of the model. We use an improved set annotations, provided every third frame (250 frames in total) and rectify the mistakes from the original annotations. Compared to PETS the sequence has similar overlap statistics than the PETS test set (see figure 2), but presents different background and motion patterns. Figure 11 shows examples from the dataset.

4.4 Pascal VOC

Pascal VOC '07 [2] is a general object detection benchmark that contains 20 classes, including people. People in this dataset are not just pedestrians but appear in many different scenes and activities. This leads to extreme variations in scale and aspect ratio. Unfortunately the dataset is not very crowded (see 2). Nevertheless we use this dataset to demonstrate how the Tnet can be adapted to handle larger variance in scale and aspect ratio.

We opt for Pascal VOC '07, because the python implementation of Faster-RCNN "py-faster-rcnn" contains a pre-trained detector for this dataset. Evaluation is done with the official Pascal evaluation toolkit, except we recompute the exact area under the curve for the AP computation as done in "py-faster-rcnn".

5 Handling scale and aspect ratio

This section explains how we adapted the Tnet to Pascal VOC to handle big scale and aspect ratio variations. The big scale differences are a problem with the fully convolutional architecture explained in the main paper, section 2, because we have a fixed convolutional filter that effectively gives the system a fixed size context to take into account. How big should this context be? In the case of relatively small pedestrians in PETS and ParkingLot it turns out that a context of 44×44 pixels is sufficient. However in Pascal two people can be almost as big as the entire image, the centre points of their bounding boxes can be several hundred pixels apart, so the context needs to be much bigger in that case.

Input grid The idea to remedy this issue is to adapt the neighbourhood size to the size and aspect ratio of the detection that is to be rescored, so big objects have a larger neighbourhood than small objects. Since we want to use the same model for big and small objects, the representation has to have the fixed size, so we use an 11×11 grid to represent the neighbourhood (defined to be twice the size of the object) just like in the ordinary Tnet. In general detections in the image have different sizes, requiring input grids of different resolutions. We decide to switch to a detection-centric representation and generate an input grid for each detection individually, which is feasible because the FasterRCNN outputs relatively few detections that were already processed individually (as opposed to fully convolutionally). The assignment of detections to the input grid is done as usual by picking that maximum scoring detection for which the centre falls into each grid cell.

Note that the network and loss is straight forward to transfer to the detection-centeric setting. The only modifications are that we used 128 filters instead of 512 filters, for speed, since we observed no noticeable drop in performance, and that we give the positive class a weight of 0.1 (compared to 0.5 on PETS), since Pascal is much less crowded.

6 DeepMask

To obtain segmentation masks on PETS, we train our reimplementation of DeepMask [11] for all classes on the COCO training set. Our implementation is based on the FastRCNN network [5]. To generate instance segmentations on PETS, we upscale the image by a factor of 2× and predict segments on detections. Figure 4 shows mask predictions for annotations on the PETS test set. It works well in low occlusion cases (left and middle column), however, under heavy occlusion it makes mistakes by collapsing the segment or merging the occluding and the occluded person (see right-most column).

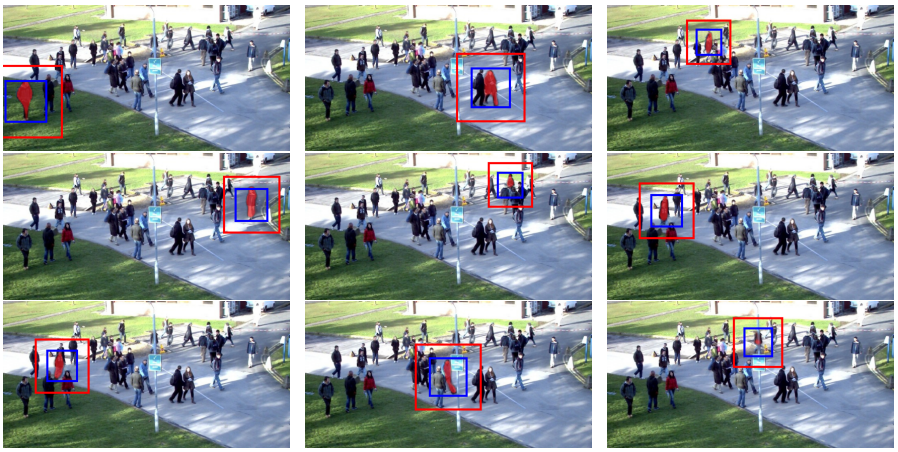


Fig. 4: Example DeepMask segmentation masks on PETS images. Pixels inside the red area are used to predict a foreground segment inside the blue area. In these examples, boxes are centred on ground truth annotations.

7 Detailed results

7.1 oMnist

Figure 5 and table 1 show the results that are analysed in section 3.1 in the main paper. Since the figure and table do not add new results, but only show an overview of the results discussed in the paper, we do not repeat their analysis here.

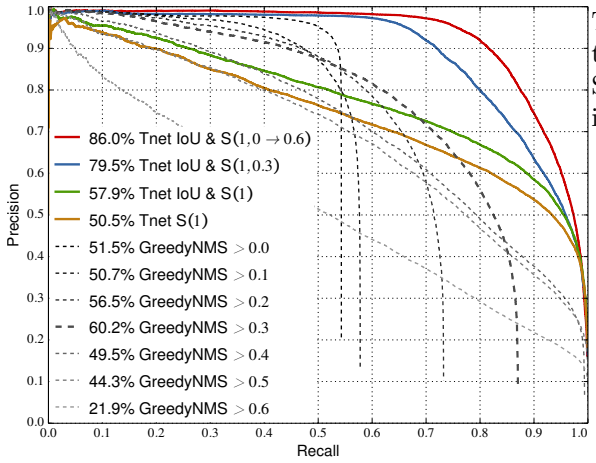


Fig. 5: Detection results on controlled setup (oMNIST test set).

Table 1: Results from controlled setup experiments. S(·) indicates the different input score maps.

Method	AR
Upper bound	
perfect masks	90.0%
GreedyNMS	
bboxes IoU > 0.3	60.2%
Tnet	
IoU & S(1, 0 → 0.6)	*86.0%*
IoU & S(1, 0.3)	79.5%
IoU & S(1)	57.9%
S(1)	50.5%

7.2 PETS

Person segments In section 4.1 of the main paper, we report results using segments estimated from the image content. We use our re-implementation of DeepMask [11], trained on the COCO dataset [9]. DeepMask is a network specifically designed for objects segmentation which provides competitive performance. Our re-implementation obtains results of comparable quality as the original; example results on PETS are provided in the appendix section 6. We use DeepMask as a realistic example of what can be expected from modern techniques for instance segmentation.

Table 2: Results on PETS validation set. Underlined is our base Tnet.

Method	AR
GreedyNMS	
bboxes IoU > 0.3	54.3%
DeepMask segments	52.0%
Tnet variants	
IoU & S(1, 0→0.6)	*59.6%*
IoU & S(1, 0.3), $r = 0.3$	58.9%
IoU & S(1, 0.3), w_g	58.0%
IoU & S(1, 0.3)	57.9%
IoU & S(1)	36.5%
S(1)	33.9%

Tuning on the validation set Validation results are shown in table 2 and figure 6. Using the relaxed loss described in section 3 helps to further improve the results. Amongst the parameters tried on the

validation set, $r = 0.3$ provides the largest improvement. Lower r values decrease performance, while higher r values converge towards the default $r = 1$ performance (base Tnet).

Weighting classes equally on the entire dataset (w_g strategy) gives a mild improvement from 57.9% to 58.0% AR compared to the default per frame weighting w_f. Using multiple GreedyNMS thresholds gives a significant improvement to 59.6% AR. We combine the weighted loss, global weighting, and multiple GreedyNMS thresholds as Strong Tnet.

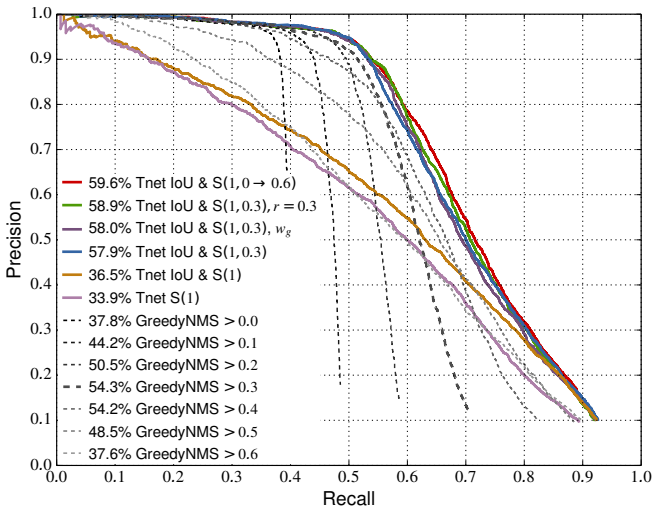


Fig. 6: Detection results on PETS validation set. Global weighting is indicated by w_g, all other curves use frame weighting.

Performance per occlusion level Figure 8 provides a more detailed view of the results from figure 7. It compares our strong Tnet result versus the upper envelope of GreedyNMS over all thresholds ([0, 1]), when evaluated over different subsets of the test set. Each subset corresponds to ground truth bounding boxes with other boxes overlapping more than a given IoU level (see figure 2). For all ranges, our strong Tnet improves over GreedyNMS. This shows that our network does not fit to a particular range of occlusions, but learns to handle all of them with comparable effectiveness.

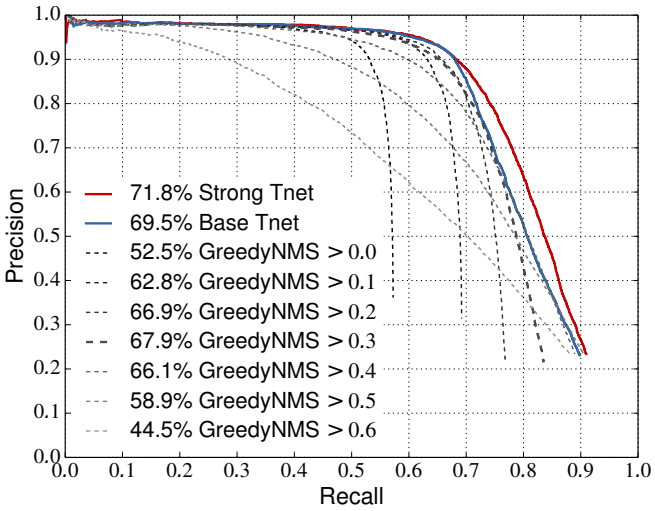


Fig. 7: Detection results on PETS test set. Our approach is better than any GreedyNMS threshold and better than the upper envelope of all GreedyNMS curves.

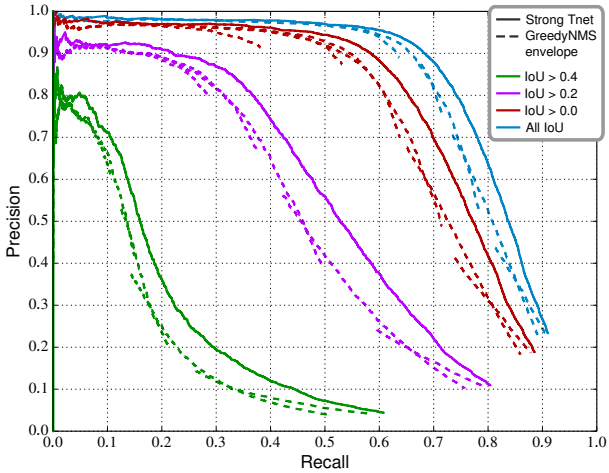


Fig. 8: GreedyNMS versus Strong Tnet when evaluated over different subsets of PETS test data, based on level of occlusion. In each subset our Tnet improves over the upper envelope of all GreedyNMS threshold curves.

7.3 ParkingLot

Figure 9 shows the curves of test results on the ParkingLot sequence with the Base and Strong Tnet that have been trained on PETS. The discussion can be found in section 4.1 in the main paper.

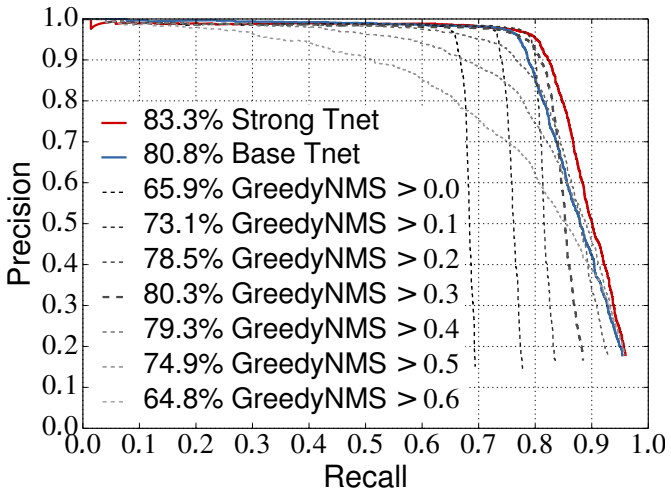
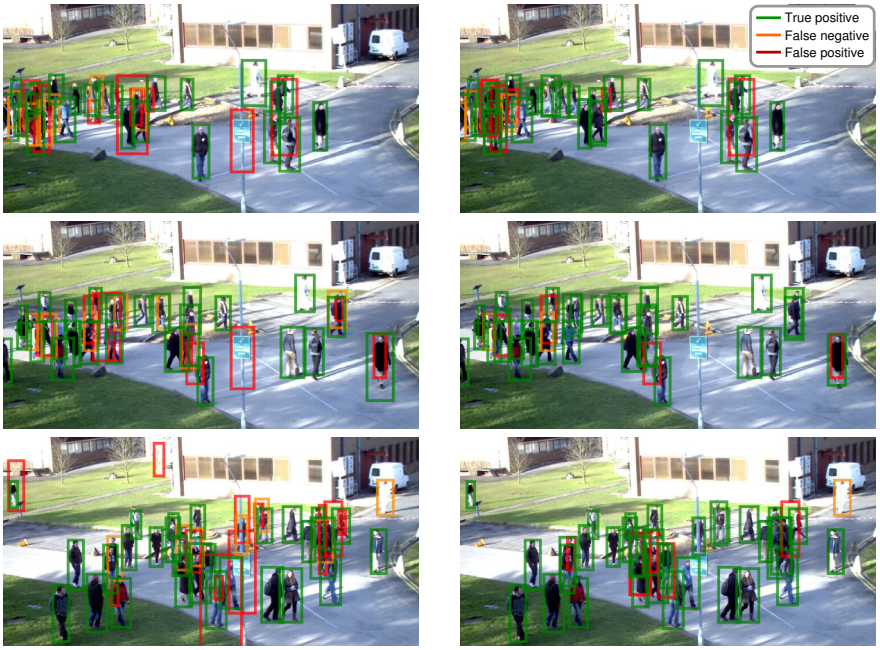


Fig. 9: Detection results on the ParkingLot dataset. Tnet is better than any GreedyNMS threshold, even though it has been trained using PETS data only.

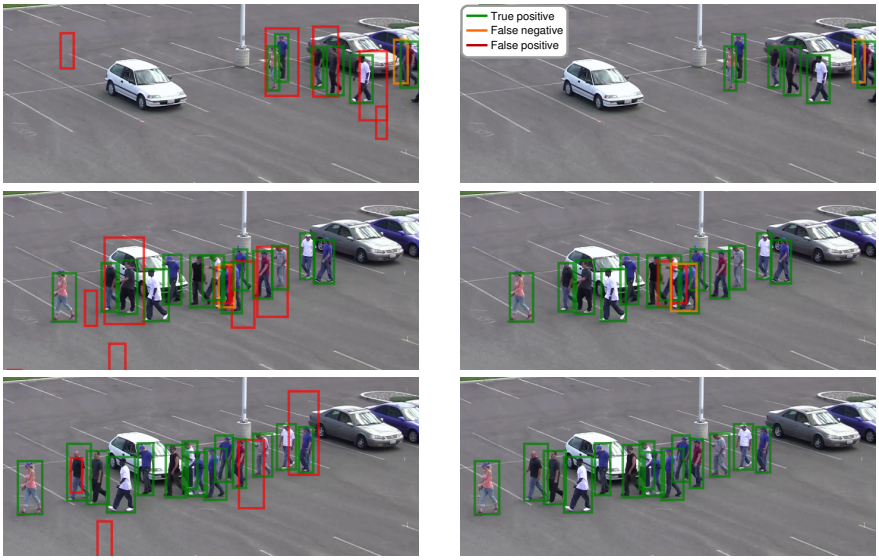
8 Qualitative results

Figures 10, 11, and 12 show qualitative results of our Strong Tnet for PETS, ParkingLot, and Pascal datasets respectively. On PETS and ParkingLot TNet provides improvement both on crowded and non-crowded areas. For Pascal we see that Tnet is able to remove implausible false positives.



(a) GreedyNMS (b) Strong Tnet

Fig. 10: Qualitative detection results of GreedyNMS > 0.3 and Strong Tnet over DPM detections on PETS test set (both operating at 75% recall). Tnet is able to suppress false positives as well as recover recall that is lost with GreedyNMS. This is the case for both crowded and non-crowded areas.



(a) GreedyNMS (b) Strong Tnet

Fig. 11: Qualitative detection results of GreedyNMS > 0.3 and Strong Tnet over DPM detections on ParkingLot dataset (both operating at 85% recall). Tnet was trained using PETS only.

(a) GreedyNMS (b) Strong Tnet

Fig. 12: Pascal persons test set examples where Tnet improves over the best GreedyNMS (IoU > 0.5). Same colour coding as fig. 10.

References

1. Dollár, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian detection: An evaluation of the state of the art. PAMI (2012)
2. Everingham, M., Eslami, S.M.A., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. IJCV (2015)
3. Ferryman, J., Ellis, A.: Pets2010: Dataset and challenge. In: AVSS (2010)
4. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR (2012)
5. Girshick, R.: Fast R-CNN. In: ICCV (2015)
6. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: ICCV (2015)
7. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: ACM International Conference on Multimedia (2014)
8. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
9. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV (2014)
10. Milan, A., Roth, S., Schindler, K.: Continuous energy minimization for multitarget tracking. PAMI (2014)
11. Pinheiro, P.O., Collobert, R., Dollar, P.: Learning to segment object candidates. In: NIPS (2015)
12. Shu, G., Dehghan, A., Oreifej, O., Hand, E., Shah, M.: Part-based multiple-person tracking with partial occlusion handling. In: CVPR (2012)
13. Subburaman, V.B., Descamps, A., Carincotte, C.: Counting people in the crowd using a generic head detector. In: AVSS (2012)
14. Tang, S., Andriluka, M., Milan, A., Schindler, K., Roth, S., Schiele, B.: Learning people detectors for tracking in crowded scenes. In: ICCV (2013)