Object Ranking on Deformable Part Models with Bagged LambdaMART

Chaobo Sun, Xiaojie Wang and Peng Lu

School of Computer, Beijing University of Posts and Telecommunications,
{cbsun, xjwang, lupeng}@bupt.edu.cn

Abstract. Object detection methods based on sliding windows has long been considered a binary classification problem, but this formulation ignores order of examples. Deformable part models, which achieves great success in object detection, have the same problem. This paper aims to give better order to detections given by deformable part models. We use a bagged LambdaMART to model both pair-wise and list-wise relationships between detections. Experiments show our ranking models not only significantly improve detection rates compared to basic deformable part model detectors, but also outperform classification methods with same features. ·

1 Introduction

Detection of objects in natural images has always been a difficult problem in computer vision, due to the complexity of backgrounds and the large variances of objects in the same category. Data-driven methods in recent years[12, 18] have achieved reasonable results. In these methods, object detection is formulated into a binary classification problem: to distinguish the object patches and non-object ones from a set of candidates. The candidates can be generated using either sliding windows[12] or objectness based methods[2, 9].

Deformable Part Models(DPM) have achieved great success in object detection. Though DPMs generate solid sets of candidates with hierarchical templates[12] to model the deformation inside categories, they also suffer from the shortcoming of binary classification formulation. Our experiments show that with an enlarged set of detections and the correct order of the detections, the performance of DPMs may be significantly prompted. This observation reveals the role of order in object detection, and make it a natural way that developing a learning to rank framework to improve DPMs.

With ranking perspective on object detection, we can remodel object detection as object ranking, which aims to recover the ground truth order given a set of candidate detections. The set, can contain all possible windows in a spatial pyramid, or the output of previous detection system.

When comparing object ranking to classical object detection methods, we find that the classical methods fail to capture the relationships between detections. The relationships, contains both pair-wise and list-wise information, plays

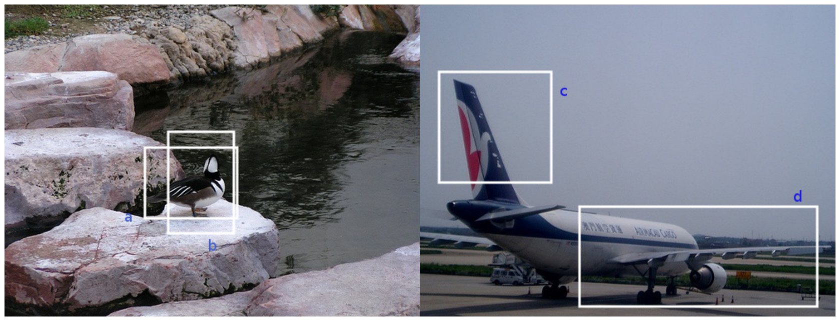

Fig. 1. In the image on the left-hand side, detection a and b are both true detections, but a is definitely a better detection than b; In the image on the right-hand side, c and d are both false detections whose overlap ratios with ground truth objects are below 0.5, but still we can tell d is better than c.

a central role in ranking. As Figure 1 shows, A ranking model focuses on "Why is a detection better than another" rather than "Why is a detection true".

We have a discussion on background works in Section 2. Section 3.1 makes a review of LambdaMART. Section 3.2 discusses why object ranking is useful. Section 3.3 describes our ranking model based on ensemble of decision trees, which uses bagged[5] LambdaMART[19], with a training procedure with increasing dataset. Experiments in section 4 discuss the features we use, the potential of DPM, show the comparison between ranking and classification. We then make analysis on the errors before and after ranking, and discuss the impact factors of our ranking model.

2 Related Work

Research on generic object detection is originating from person detection[10]. From then on sliding-window methods with HOG pyramids have been a main stream on object detection. For every category of objects, sliding-window build a set of templates to represent all its poses. During training, cropped objects and backgrounds are extracted to train the template. During detecting, a matching score is computed at every position in the feature space, then the position with scores above a threshold is considered to be an object position[10].

Deformable part models[12] have greatly pushed the research on object detection. As a variation of sliding windows, DPMs establish a set of hierarchical templates for every category of objects. Each template is organized into a root and its parts. Not only the appearance(vision features) of roots and parts, but also the parts' relative positions(structural features) to the root are taken into consider, so that DPMs can tolerate a certain degree of deformation.

Our work is basically based on DPM. But different from the way using strong supervised information[4, 3], we aim to improve DPM by using extra image information beyond HOG, which is relatively simple.

Many excellent works have shown that object detection can benefit from objectness models[2, 18, 9]. Objectness models aim to provide a small set of detection candidates with high recall. Then classifiers are applied on the candidates. Our work is similar to this approach in the way that we both use two-stage modeling, but we use deformable part models rather than objectness models to generate more reliable, and smaller sets of candidates, also we use ranking rather than classification as the post procedure.

Gradient Boosted Decision Trees(GBDT) based methods have achieved great success in learning to rank problems[6]. They use a pair-wise loss from ranknet models[7]. GBDTs are then extended to LambdaMART[19] by introducing list-wise information[8]. We use a bagged LambdaMART to rank detections, this method is similar with [13], but differs in the way how bagging samples are generated, our generating scheme is based on pairs rather than examples, and is a more natural way in the task of object detection.

There are many types of useful image features for object detection, among which Histograms of gradient(HOG) features[10] have long been a primal one, Local Binary Patterns(LBP)[1] are proved to be a good supplement for HOG, Color SIFT has also been a standard feature using in methods based on selective search[18], local contrast information and saliency are also useful[2].

In recent years features from deep convolutional neural networks(CNN)[15] achieved very high performance[14]. Features learned by well-structured deep CNNs are good descriptors for images, and are powerful tools for object detection.

3 Ranking Model for Object Detection

3.1 Why Ranking?

Classical learning to rank systems focus on selecting relevant items from a set of candidates. Object detection is similar to these models, if we interpret the searching space of object detection as a set of candidates. Following the way of Pascal VOC evaluation[11], the set of candidates are all positions of all images in a dataset. The aim of object detection is then selecting candidates that have more overlap ratios with ground truth objects.

In information retrieval systems, when modeling the relationship of some samples $\{x_i\}_{i=1}^n$ and their corresponding labels $\{y_i\}_{i=1}^n$, there are three types of information:

- item-wise information, direct relationships between every x_i and y_i.
- pair-wise information, relationships between a paired (x_i, x_j).
- list-wise information, information retrieval metrics of a list $x_1, x_2, \ldots, x_n$, such as mean average precision(MAP).

On the other hand, the standard evaluation metric for object detection is MAP, so it is natural that taking MAP of detections into account during training. While it is quite difficult for classification which focuses on item-wise labels, adding list-wise information to a ranking model is straight-forward[8].

The key difficulty for applying ranking models on object detection is its large space of candidates. All rectangles in images are candidates to rank. Sliding window methods largely reduce the number of candidates by making the constraint that all candidates should be in certain sizes[10], while in recent years there are several useful technologies directly aiming at shrinking the space of candidates[2, 9].

3.2 LambdaMART

Cross Entropy Loss Suppose that we have a list of detections, labeled with their overlap ratios with ground truth objects. We first generate a set of pairs based on the list, let the set be J, each pair (i, j) in the set J means that detection x_i has a higher overlap ratio than x_j with some ground truth objects. We then define a cross entropy loss function on pairs in J.

To begin the definition, we define a simple empirical distribution on every pair (i, j) in J:

$$\bar{\mathrm{P}}_{\mathrm{ij}} \equiv \begin{cases} 1, \ (i,j) \in J \\ 0, \ (j,i) \in J \end{cases} \tag{1}$$

The empirical probability is a statistical measure of the pair-wise information in training datasets. During the train stage of our model, the score function is applied on each detection, and the score outputted would also generate a model distribution. We define it in a form of sigmoid function:

$$P_{ij} \equiv \frac{1}{1 + e^{-\sigma(f_i - f_j)}} \tag{2}$$

For simplicity, we use f_i to denote a score function for sample x_i.

The two distributions should be as close as possible. We use cross entropy to measure the divergence of them:

$$C_{ij} = -\bar{P}_{ij} log(P_{ij}) - (1 - \bar{P}_{ij}) log(1 - P_{ij}) \tag{3}$$

Combining Eq.(1) ,(2) and (3), we got:

$$C_{ij} = log(1 + e^{-\sigma(f_i - f_j)}) \tag{4}$$

MART A MART model is an ensemble of regression decision trees, the score for a feature x is defined as:

$$f^m(x) = \sum_{k=1}^{m} \eta^k t^k(x) \tag{5}$$

where $t^k(x)$ is the score of kth regression decision tree, η is the learning rate.

Suppose that $m - 1$ trees are trained, and the mth tree is now to be trained. Note that in MART model, a new regression tree tries to capture the gradient of total cost, that is, let the gradient for a single example x_i be λ_i^m, the mth tree trained to fit the dataset $\{(x_i, \lambda_i^m)\}_{i=1}^k$. We have:

$$\lambda_i^m \equiv \sum_{j:(i,j)\in J or (j,i)\in J} \frac{\partial C_{ij}^{m-1}}{\partial f_i^{m-1}} \tag{6}$$

Recall the derivative of C_{ij}, which is defined in Eq.4, has following feature:

$$\frac{\partial C_{ij}}{\partial f_i} = \frac{-\sigma}{1+e^{-\sigma(f_i-f_j)}} = -\frac{\partial C_{ij}}{\partial f_j} \tag{7}$$

let $\frac{\partial C_{ij}^{m-1}}{\partial f_i^{m-1}}$ be λ_{ij}^m, we can re-write λ_i^m as:

$$\lambda_i^m = \sum_{j:(i,j)\in J} \lambda_{ij}^m - \sum_{j:(j,i)\in J} \lambda_{ij}^m \tag{8}$$

LamdbaMART While the above formulation well models pair-wise information, weights of λs are introduced to capture list-wise information.

$$\lambda_{ij} = \frac{-\sigma}{1+e^{-\sigma(f_i-f_j)}} \|\Delta MAP_{ij}\| \tag{9}$$

where ΔMAP_{ij} is the change of mean average precision if positions of x_i and x_j are exchanged.

3.3 Bagged LambdaMART with an Increasing Training Set

We have discussed in Section 3.1 why object detection is more a ranking problem instead of classification, but there are still significant differences between object detection and classical ranking problems.

The first difference is that the list of candidates in object detection is much larger. In classical ranking tasks, candidates are divided into different groups, because only candidates in the same group have relationships, comparison between groups are meaningless. While in object detection, a candidate is comparable with all other candidates in the dataset, not only from the same image, so an ideal list for object detection is a list contains all candidates detected on all images in the dataset.

Suppose there are m images in the dataset, and we generate n detection candidates per image, then the ranking list has length $m*n$, and when applying LambdaMART in which pairs are needed to be generated, that is, $O((m*n)^2)$ pairs(if the relationships between candidates are not sparse, they are usually not), computing the loss of all pairs are both time consuming and memory consuming.

The second difference is that there are too many low-ranking candidates relative to high-ranking ones. This is a main difficulty for nearly all object detection

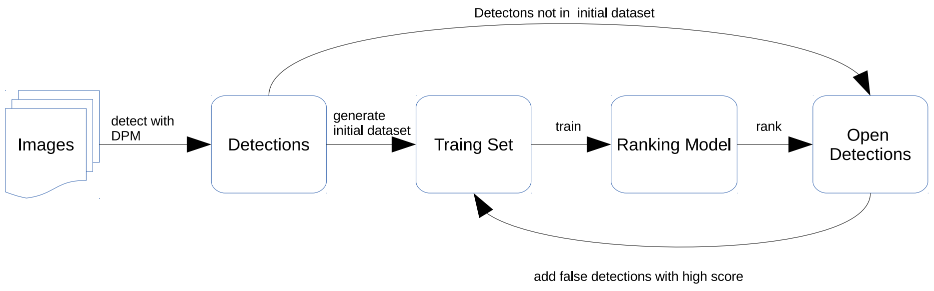

Fig. 2. The training procedure

systems[12, 20, 18]. DPM results have largely reduced the difficulty by providing a credible candidate sets, but we still have a ratio of $25:1$ between high-ranking detections with low-ranking ones.

Based on above observations, we use an ensemble of ranking models with increasing training set for detection. The framework of training is shown in Fig 2.

Increasing Training Set We generate the training set in an increasing way. The training set is initialized with good candidates whose overlap ratios are above a threshold, others are stored in a set of **open detections**, trained models are applied on this set and detections with high scores are added into the training set and removed from the set of open detections. Operating like this iteratively, we are supposed to get final model after a certain number of iterations.

Increasing training datasets, which are widely used in object detection systems, aim to solve the unbalance problem of true and false examples. Our implementation is similar to previous methods based on classification, but the motivation is quite different.

It is notable that in the ranking perspective on object detection, there is no unbalance problem for training data, as we focus mainly on pair-wise and list-wise information, rather than item-wise labels. So it is possible to include all examples in training set, and train a model in a single round, our iterative setting is aiming to reduce the difficulty in computing.

Bagging At each iteration, once training set is prepared, the training set forms a single list for ranking. To reduce computational complexity, we use bagging methods to randomly split the whole set into smaller subsets. A splittings is a bagging[5] set of samples, in the way that splitting on examples are equivalent to sampling on pairs.

To ensure pairs are sampled uniformly, suppose we have K subsets, and the probability of sample x_i be grouped into subset $k(1 \leq k \leq K)$ is p_i^k, then the probability of pair (i, j) be sampled in this splitting is $\frac{\sum_{k=1}^{K} p_i^k \cdot p_i^k}{K}$. We can then

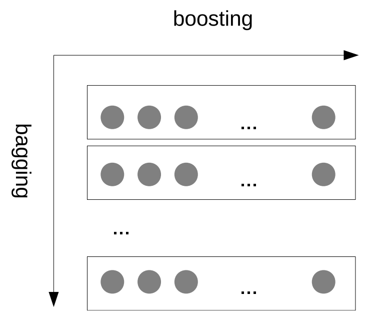

Fig. 3. Ensemble of decision trees as a matrix

make probability a constraint by simply setting $p_i^k = \frac{1}{K}$. That means if each example is assigned into subsets uniformly, then the pairs are sampled uniformly.

We then train a LambdaMART model for every splitting, then the ensemble of LambdaMART models are obtained as the final model at this iteration.

Bagged LambdaMART can be interpreted as a matrix of decision trees, as Figure 3 shows, trees in the model are ensembled in both gradient boosted scheme, and bagging scheme.

4 Experiments

4.1 Deep features

We use Caffe[16] to extract deep features. The network is the pretrained model in Caffe. The architecture of the neural network, is defined in [17]. For each detection windows, the image patch bounded by it are feed into the network, and the output of penultimate layer are taken as features. These features have been proved to be powerful in object-based image recognition problems[14], and it would be more powerful after fine-tuning on Pascal VOC datasets, but as our focusing are the ranking procedure, we just use the original features.

4.2 Potential of DPM

Deformable Part Models, which learn certain number of templates for objects, suffer from the large variance of objects in a category, but we argue that by enlarging the detection set on each image, and giving them the correct order, deformable part models can achieve much better performance.

We enlarge the detection set by selecting k-best detections per image, instead of using a static threshold. Fig.4 shows that under k-best scheme, recall is an increasing function on k.

To illustrate the importance of correct order, we evaluate detections in both orders: the order given by DPM detectors, and the order given by their overlap ratios.

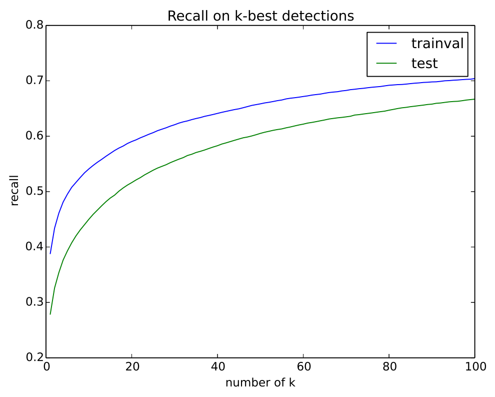

Fig. 4. recalls are growing by increasing k on both trainval and test dataset of Pascal VOC 2007, but the growth would be very slow when k is large.

Table 1 shows the results of comparison on Pascal VOC2007 datasets[11], which is also the potential of DPM.

Table 1. DPM order is the order according to DPM scores, and GT order is the order according to overlap ratios with ground truth objects. When detections are given, DPM results in GT order are the upper bound of DPM.

	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	
DPM order	0.310	0.597	0.040	0.121	0.235	0.506	0.546	0.171	0.177	0.228	
GT order	0.748	0.875	0.700	0.609	0.602	0.879	0.710	0.912	0.755	0.698	
	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	AVG
DPM order	0.221	0.046	0.583	0.479	0.418	0.085	0.188	0.359	0.454	0.408	0.309
GT order	0.898	0.893	0.876	0.845	0.691	0.662	0.641	0.960	0.868	0.785	0.780

4.3 Model Comparison on Pascal VOC Benchmarks

While correct order can significantly improve the performance of DPM, how to recover the correct order is a main challenging. As discussed above, ranking is a straight-forward way. We also implement a binary classification re-scoring procedure based on SVM for comparison.

Table. 4.3 shows that our ranking model outperforms classification model based on SVM. The results proved our guess: by modeling more information, ranking models are more powerful than classification models in object detection.

In table 4.3 we make an extra comparison between our results and [14], which uses the same features, much larger candidate sets(1000-2000 detections per image, giving higher recall than DPM), and svm classifiers. With a smaller

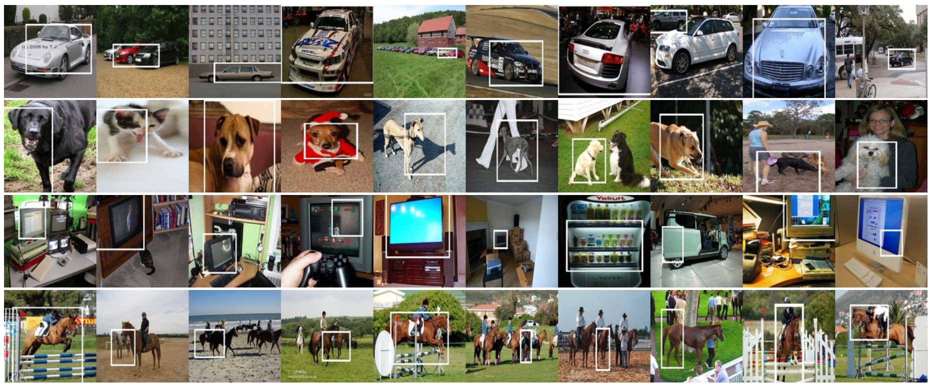

Fig. 5. Top detections of car, dog, tvmonitor and horse, which are considered negative by original DPM, but judged positive by our ranking model.

set of candidates, our models plays better on 9 categories, and average rate on all categories are very close(only a difference of 0.3

The implementation in [14] uses training and testing set of detections generated by selective search, which contains 1000-2000 detections per image. We obtain close performance to them with a smaller set of candidates.

Table 2. DPM is the original results, DPM+svm is the results, DPM+rank is results form our ranking model, DeepF is the result reported in [14].

	plane	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv	AVG
DPM	0.310	0.597	0.040	0.121	0.235	0.506	0.546	0.171	0.177	0.228	0.221	0.046	0.583	0.479	0.418	0.085	0.188	0.359	0.454	0.408	0.309
DPM+svm	0.421	0.555	0.341	0.194	0.261	0.519	0.545	0.428	**0.327**	0.291	0.391	0.337	0.501	0.591	0.417	0.192	0.313	0.329	0.451	0.477	0.394
DeepF	**0.531**	0.589	**0.354**	**0.296**	0.223	0.5	**0.577**	**0.524**	0.191	**0.435**	**0.408**	**0.436**	0.476	0.54	0.391	**0.23**	**0.423**	0.336	0.514	**0.552**	0.426
DPM+rank	0.467	**0.668**	0.259	0.194	**0.307**	**0.594**	0.570	0.392	0.291	0.365	0.335	0.240	**0.651**	**0.629**	**0.445**	0.167	0.309	**0.449**	**0.586**	0.541	0.423

Fig.5 shows top detections from negatives of DPM. The ranking model are more strong at detecting truncated objects or objects in complex backgrounds.

4.4 Analysis of Ranking Model

Following [18], we initialize the training set with detections whose overlap ratios are above 0.2. At each iteration, 5000 detections with highest score in the set of open detections are added into the training set.

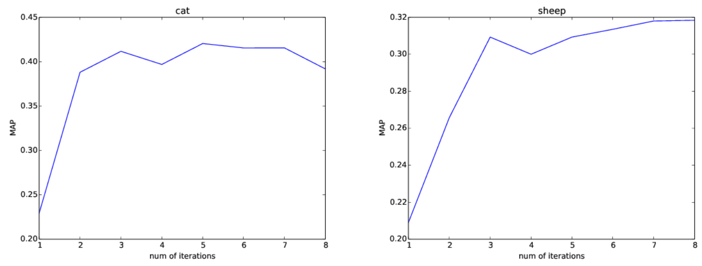

Fig. 6. We use two categories to select a best number of iterations. We train ranking models on trainval dataset of Pascal VOC 2007, and test them on the a subset of the test dataset.

Fig.6 shows detection rates at different iterations. The first three rounds of training is useful, but training after 3 rounds makes little contribution. At the beginning iterations of training, MAP grows significantly, but after 3 rounds, there is no significant promotion when increasing the number of iterations.

4.5 Analysis of Errors

Under the standard evaluation metric of MAP, true detections are the detections which have overlap ratios below 0.5, and false detections just the opposite. When focusing on false detections, it is a natural idea that these errors are caused by different reasons, so that a fine-grained analysis is reasonable.

We divide detection errors into 3 types: location errors; confusion with other categories; background errors. We argue that only background errors are *real* errors. Location errors at least provides the correct information about the existence of objects, while confusion errors may suggest visual similarities of different categories.

We define location errors as the false detections with relatively high overlap ratios(above 0.2). As Fig.7 shows, giving the same number of total errors, the proportion of location errors after ranking are much higher than that before ranking.

The result means that false detections with higher overlap ratios are given higher ranking, and the our ranking model has a strong ability of distinguishing a not so bad example from totally bad ones. Then, by introducing pair-wise and list-wise information, the ranking model meets the expectation that learning "why a detection is better than another".

Another important type of errors is confusion with other categories. Fig.8 shows the confusion matrices on top 100 errors before and after ranking.

Overall distribution of confusions before and after ranking are similar. There are some common pairs in both matrices, like **bicycle** and **motorbike**; **car** and **bus**; **horse** and **cow**; **train** and **bus**. These pairs are all similar categories in vision, it is difficult to distinguish even by human visually, human may distinguish them by other information.

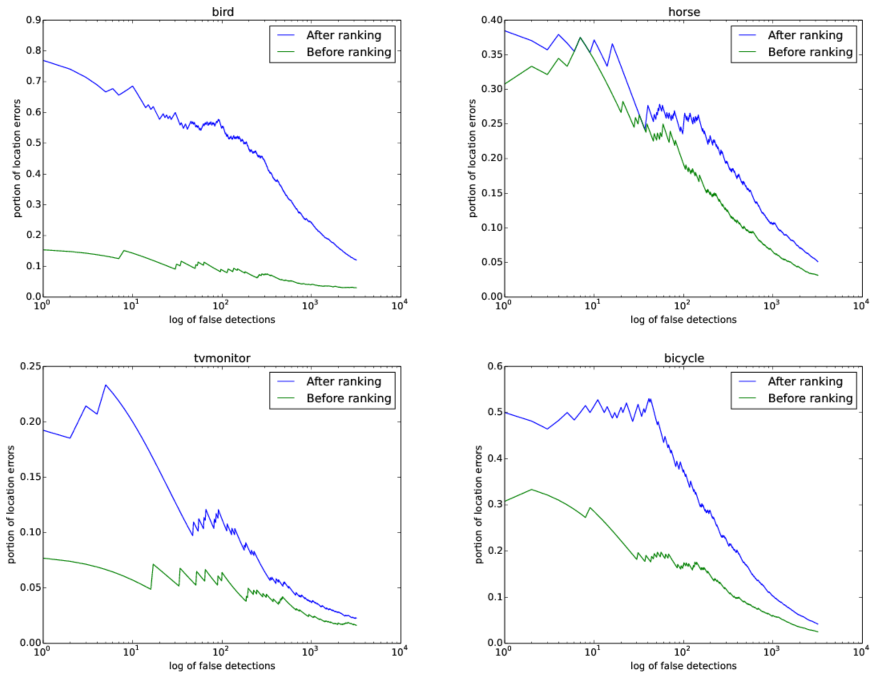

Fig. 7. We select 4 representative classes: bird, on which the performance of DPM is very poor; horse, on which the performance of DPM is very good; tvmonitor and bicycle are artificial objects with middle level performance.

Besides similar pairs discussed above, it is interesting that there are two significant confusion in the matrix before ranking: recognizing bird as aeroplane, and recognizing person as chair. They are both dismissed in the matrix after ranking. It shows that when HoG features focus on the shape and structure of images, deep features are more complex and more powerful to distinguish images by appearance.

5 Conclusion and Future Work

Our whole work in this paper is motivated by the ignorance of order information by classical object detection methods. Under the guidance of this idea, we propose a bagged LambdaMART for object detection. We evaluate the models on k-best results generated by deformable part models, with deep features from convolutional neural network. Experiments show an improvement by ranking not only compared to original DPM, but also svm-based re-scoring method applied on DPM. Our model also achieves close result with state of the art methods which use the same feature. We also implement several fine-grained evaluations

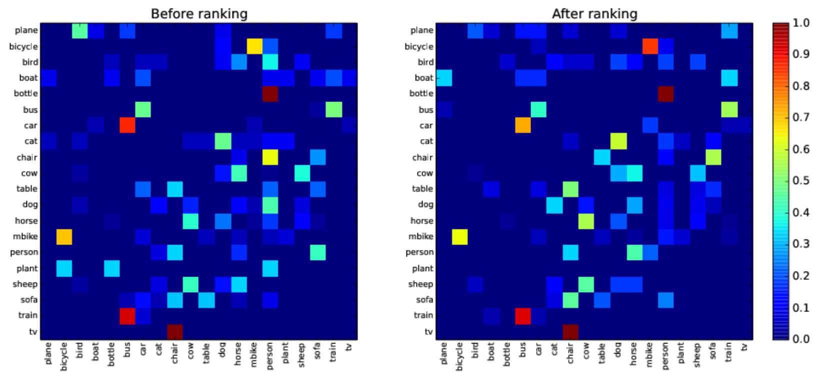

Fig. 8. Confusion matrices before and after ranking. These matrices are not standard confusion matrices, only errors are shown in them so that the diagonals are all zero. Each row in the matrices represents distribution of mis-recognition errors for a specific detector.

on detection errors, which also provides solid evidence for the role of ranking in object detection.

But it is also notable that the combination of ranking model and deep features does not explore total potential of DPM, there is still a long way to go to recover the ground truth order.

Our object ranking framework can be used as a post-processing stage of any object candidates generating system. Although the k-best results of DPM are small and reliable, there are many objectness based methods which generate results with much higher recall. It would be also useful to run the ranking model on that methods.

Acknowledgments. This work was partially supported by National Natural Science Foundation of China(Project No.61273365 and No. 61100120), the Fundamental Research Funds for the Central Universities (No. 2013RC0304), National High Technology Research and Development Program of China(No. 2012AA011104) and discipline building plan in 111 base(No. B08004) and Engineering Research Center of Information Networks, Ministry of Education.

References

1. Timo Ahonen, Abdenour Hadid, and Matti Pietikinen. *Face Recognition with Local Binary Patterns.* 2004.
2. Bogdan Alexe, Thomas Deselaers, and Vittorio Ferrari. What is an object? In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 73–80. IEEE, June 2010.
3. Hossein Azizpour and Ivan Laptev. Object detection using strongly-supervised deformable part models. *csc.kth.se*, 2012.

4. Steve Branson, Pietro Perona, and Serge Belongie. Strong supervision from weak annotation: Interactive training of deformable part models. *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1832–1839, November 2011.
5. Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.
6. Chris Burges. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11:23–581, 2010.
7. Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, pages 89–96. ACM, 2005.
8. Christopher JC Burges, Robert Ragno, and Quoc Viet Le. Learning to rank with nonsmooth cost functions. In *NIPS*, volume 6, pages 193–200, 2006.
9. Ming-Ming Cheng, Ziming Zhang, Wen-Yan Lin, and Philip H. S. Torr. BING: Binarized normed gradients for objectness estimation at 300fps. In *IEEE CVPR*, 2014.
10. Navneet Dalal, Bill Triggs, and De Europe. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. Ieee, 2005.
11. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.
12. Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1627–1645, 2010.
13. Yasser Ganjisaffar, Rich Caruana, and Cristina Videira Lopes. Bagging gradient-boosted trees for high precision, low variance ranking models. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, pages 85–94. ACM, 2011.
14. Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *arXiv preprint arXiv:1311.2524*, 2013.
15. Kevin Jarrett, Koray Kavukcuoglu, M Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2146–2153. IEEE, 2009.
16. Yangqing Jia. Caffe: An open source convolutional architecture for fast feature embedding. http://caffe.berkeleyvision.org/, 2013.
17. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.
18. Koen EA van de Sande, Jasper RR Uijlings, Theo Gevers, and Arnold WM Smeulders. Segmentation as selective search for object recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1879–1886. IEEE, 2011.
19. Qiang Wu, Chris JC Burges, Krysta M Svore, and Jianfeng Gao. Ranking, boosting, and model adaptation. *Tecnical Report, MSR-TR-2008-109*, 2008.
20. Long Zhu, Yuanhao Chen, Alan Yuille, and William Freeman. Latent hierarchical structural learning for object detection. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1062–1069. IEEE, 2010.