Saliency-guided Selective Magnification for Company Logo Detection

Christian Eggert, Anton Winschel, Dan Zecha, Rainer Lienhart
Multimedia Computing and Computer Vision Lab
University of Augsburg
Augsburg, Germany
Email: {christian.eggert, anton.winschel, dan.zecha, rainer.lienhart} @ informatik.uni-augsburg.de

Abstract—**Fast R-CNN is a well-known approach to object detection which is generally reported to be robust to scale changes. In this paper we examine the influence of scale within the detection pipeline in the case of company logo detection. We demonstrate that Fast R-CNN encounters problems when handling objects which are significantly smaller than the receptive field of the utilized network. In order to overcome these difficulties, we propose a saliency-guided multiscale approach that does not rely on building a full image pyramid. We use the feature representation computed by Fast R-CNN to directly classify large objects while at the same time predicting salient regions which contain small objects with high probability. Only selected regions are magnified and a new feature representation for these enlarged regions is calculated. Feature representations from both scales are used for classification, improving the detection quality of small objects while keeping the computational overhead low. Compared to a naive magnification strategy we are able to retain 79% of the performance gain while only spending 36% of the computation time.**

I. Introduction

Object recognition has made major advances since the advent of deep convolutional neural networks. With easy access to large datasets and increasing accuracy and speed of deep learning methods, commercial interest to exploit these methods also increases. Company logo detection is one of such business interests.

On the surface, company logo detection is nothing but a special case of general object detection. However, there exist some subtle differences: Unlike general object detection which has to account for deformable objects and a huge variety of potential appearances, company logos are comparatively rigid in appearance and are typically found on planar surfaces which tends to simplify the detection.

Other factors tend to complicate the detection task. One such factor is the typical size of the object instance compared to the size of the image. Most datasets for object detection, such as VOC2007 [1] and MSCOCO [2] consist of comparatively small images. Objects typically occupy a rather large area compared to the size of the image. For many applications it is not necessary to detect every small object instance. In order to analyze a scene it is often only necessary to detect the most prominent objects.

By contrast, company logos tend not to be the primary object the photographer intended to depict and just happen to get recorded in the background along with the scene. This means that company logo detection often has to deal with high-resolution images whose objects usually occupy only a small fraction of the image area.

In this work we examine some of the problems which can arise when using the well-know Fast R-CNN approach to detect such objects. Our contributions are as follows: (1) We supply updated annotations and evaluation tools for the task of object detection on FlickrLogos-32 [3], a well-known dataset for logo retrieval and classification. (2) We show that small objects are generally detected with a lower confidence, leading to a poor detection performance, especially in high-precision applications. (3) We propose and evaluate a multiscale detection approach which does not rely on building a full image pyramid.

II. Related Work

R-CNN [4] is a well-known approach for object detection with deep convolutional neural networks (DCNNs). It makes use of network architectures like AlexNet [5] or VGG16 [6] which initially were conceived for classification tasks. A common trait among those architectures is an input layer of fixed dimensions.

Within the R-CNN framework, regions of interest (ROIs) are identified in an input image. For this purpose, object proposal algorithms like selective search [7] or edge boxes [8] are used. Each ROI is resized to fit the input window of a DCNN and is classified separately. Since the evaluation of a DCNN is a computationally intensive task, this process can become very slow, depending on the number of ROIs in the image.

Fast R-CNN tries to speed up this evaluation by taking advantage of two properties: (1) The ROIs typically overlap to a large degree (2) The convolutional layers are agnostic to the input size.

Fast R-CNN takes two inputs: An image and a list of ROIs. The convolutional layers are easily resized to fit the dimensions of the input image. On the other hand, the fully connected layers require an input of fixed size. In order to solve this problem a ROI-Pooling layer is introduced which takes as input the feature map of the complete image and the list of ROIs. For each ROI a corresponding region on the feature map is calculated and the features are pooled into a fixed-size representation using max-pooling. The now fixed-dimensional feature representation for each ROI is then fed

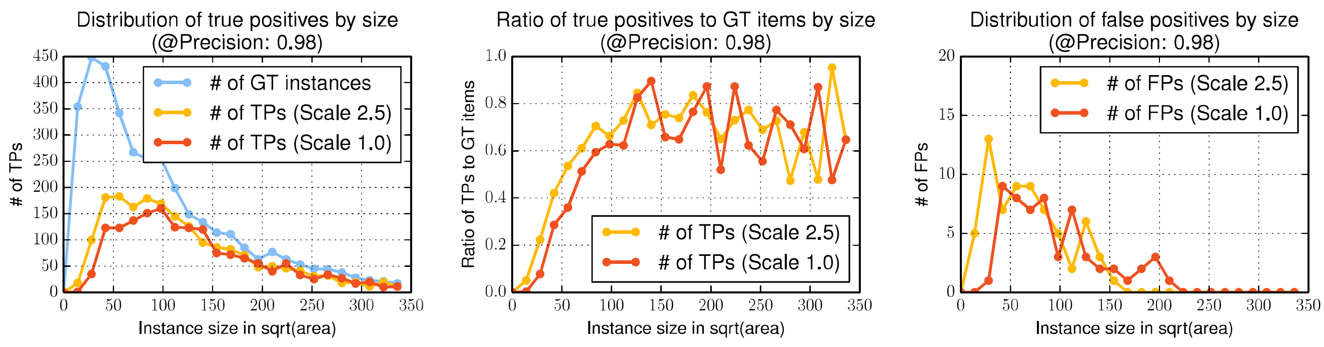

Fig. 1. Distribution of groundtruth items, true positives and false positives on the FlickrLogos dataset for a given precision of 0.98 using Fast R-CNN trained on two different scales.

into the fully connected layers for classification. Fast R-CNN can therefore avoid to compute a new feature representation for each ROI which would result in much redundant computation due to the overlapping nature of the ROIs.

A further improvement to this approach is Faster R-CNN [9] which does not require externally supplied ROIs but generates them by means of a region proposal network. However, after these ROIs are generated, the classification pipeline is identical to Fast R-CNN. In this work we focus solely on the feature representation associated with the ROIs. As a result we do not concern ourselves with the source of the region proposals. For the scope of this work we can regard both approaches as roughly equivalent and most results obtained for Fast R-CNN should also apply to Faster R-CNN.

III. Fast R-CNN on small objects

While Fast R-CNN offers a great speedup compared to R-CNN, the two approaches are not equivalent. The output neurons of the convolutional layers of a DCNN typically have a very large receptive field. In the widely used CaffeNet [10] layout for example, every neuron after the last max-pooling operation has a receptive field of 195 pixels. For VGG16 [6] – another widely used layout – the receptive field for the equivalent neurons is even larger, covering 212 pixels. In both network layouts, the receptive fields of two neighboring neurons behind the last pooling layer differ only by 32 pixels. Small objects therefore occupy only a small fraction of the receptive field, yet they can be perceived by multiple neurons. This means that in order to to extract meaningful features, the network has to learn to respond to very localized stimuli while suppressing most parts of the receptive field.

In the R-CNN framework, small objects manifest themselves in a different way. Since all ROIs – no matter how small they are – are rescaled to a fixed size, small objects will appear blurred in the input window of the network but always filling the receptive field of the central neuron of the feature map.

We argue that the input size plays an important role in the DCNNs ability to extract distinctive features which in turn can improve the detection performance. In order to demonstrate this property, we evaluate the detection performance of Fast R-CNN on two differently scaled versions of the FlickrLogos dataset.

We use region proposals generated through a modified [11] version of selective search [7] which improves the identification of text-based objects. Since selective search is biased towards large objects, we use a rather larger number of proposals – up to 8000 bounding box proposals per image – to minimize the exclusion of small objects at the proposal stage. As long as the object proposals are able to identify small object candidates, the nature of the proposals is not particularly important for our analysis since we are only interested in the suitability of the corresponding feature representations.

We train and evaluate Fast R-CNN on the original images (scale 1.0) and magnified versions (scale 2.5) using the same object proposals for both scales. In doing so, we notice an increase in mAP from 0.668 to 0.711. In Table I we provide APs for each individual class while Figure 1 shows the distribution of true positives and false positives as a function of the object size. In order to generate this figure, a precision of 0.98 has been chosen. Figure 1 makes it clear that the increase in performance is due to an improved detection of small objects and that Fast R-CNN has difficulties detecting small objects in such a high precision setting. This also suggests that small object instances are typically detected with a lower score than large object instances. By scaling up the input images this problem could be mediated, enabling the detection of smaller objects with a higher confidence.

Note that object instances with a side length smaller than 120 pixels consistently benefit from an upscaled input image while for object instances with a side length larger than 120 pixels the detection performance tends to decrease. We argue, that this is due to the fact that by upscaling the image, large object instances are magnified beyond the 212 pixel receptive field of the VGG16 network.

These experiments suggest that while convolutional neural networks are indeed fairly insensitive to scale, they tend to work best when objects have approximately the same size as the receptive field. For completeness, we have also plotted the number of false positives by size. It can be seen from the plot that small logo instances tend to be misclassified more often than large instances but the total number of false positives remains largely unaffected by the magnification.

In order to exploit this property for datasets containing large scale variations, this problem could in theory be mediated by constructing an image pyramid and running the detection on multiple scales. However, both memory requirements as well as computational complexity increase quadratically with the scaling factor. Therefore, this approach is only viable in cases where objects occupy a large area of the image, relative to the image size – which tends not to be the case for company logo recognition. For comparison, the size of the average object instance in the popular VOC2007 test set is 33% of the image side length (measured in $sqrt(area)$) while for the FlickrLogos test set it is only 15%.

IV. Selective Magnification

The computational costs of applying a DCNN lead Girshick et. al [4] to turn away from a traditional sliding window approach for object detection and towards object proposal algorithms which allow to evaluate only selected regions of interest. Similarly, we propose to only selectively magnify image regions which are deemed interesting enough to warrant closer examination.

In order to select ROIs that are likely to contain interesting structures we train an SVM as a binary classifier to predict the presence or absence of an object using the feature representation that Fast R-CNN provides for each ROI. Typically, a classifier for object detection has to be quite powerful. It has to provide a localized response only at object locations and avoid false detections while at the same time being able to distinguish multiple object classes from each other. Since the objective of our classifier is to only perform a pre-selection of interesting bounding boxes, we do not necessarily need a classifier with high precision. Therefore the main requirement is to reduce the number of bounding boxes to be tested while not missing any interesting objects. A SVM seems to be a suitable choice for this task since we only have a binary classification problem and are able to easily adjust the threshold to achieve the desired recall.

We postulate that this is a much easier task for a classifier than object detection and we argue that this task can be accomplished with low-quality features as obtained by Fast R-CNN on small objects. Figure 2 demonstrates the viability of this approach on the FlickrLogos dataset, showing the percentage of bounding boxes classified as positive as a function of the recall. For this experiment we have selected small object instances with a side length between 15 and 80 pixels as well as object proposals with an overlap (as determined by intersection over union) of at least 0.5. Negative examples are obtained by sampling selective search boxes which do not have sufficient overlap with a groundtruth item. We use features from Fast R-CNN obtained by 1×1 and 3×3 ROI-Pooling to train an SVM which is evaluated on corresponding features from selective search bounding boxes on the test dataset. An object proposal is counted as a positive example if it possesses an intersection over union of at least 0.5 with a groundtruth item.

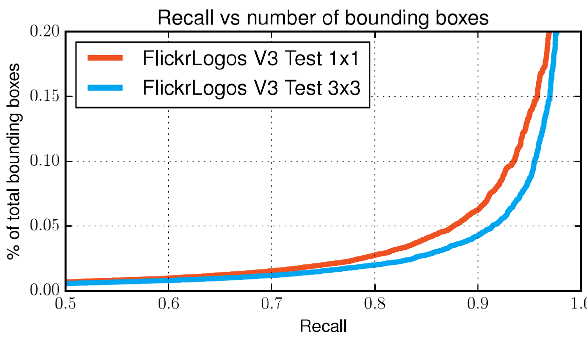

Fig. 2. Relative number of bounding boxes which are deemed likely to contain objects by an SVM classifier as a function of the Recall. Only a fraction of bounding boxes need to be examined more closely. Shown is the performance of features obtained by 1x1 and 3x3 ROI-Pooling.

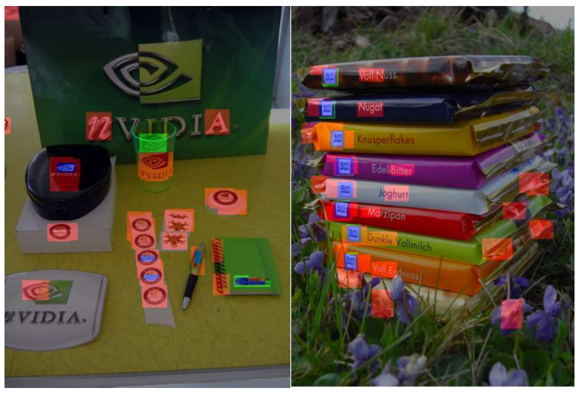

Fig. 3. Examples for ROIs selected for magnification. Selected ROIs are distributed sparsely across the image. (red) Image areas selected for closer inspection. (blue) Selected ROIs which are overlapping with groundtruth annotations for small objects (side length $\leq$ 80 px). (green) Groundtruth annotations for small objects not covered by any detection.

Figure 2 clearly shows that the classifier is able to predict the presence of small object instances quite well while excluding a large portion of possible ROIs. For example we can achieve a recall of more than 80% while only retaining 5% of the bounding boxes for closer inspection. Seeming like a reasonable choice we select 5% of the bounding boxes as an operating point for further experiments and adjust the threshold for the SVM accordingly.

Figure 3 shows example images containing a visualization of the areas which are selected by the SVM for closer inspection while Figure 4 shows the ROIs contained within these selected areas. From these two images we can make three observations: (1) The image area occupied by ROIs classified as positive is small compared to the full image. (2) ROIs which are classified as positive are not distributed evenly over the image but tend to appear in clusters. (3) Image areas which are deemed interesting by the SVM still contain many overlapping ROIs, despite their small scale.

Taken together, these observations mean two things: (1)

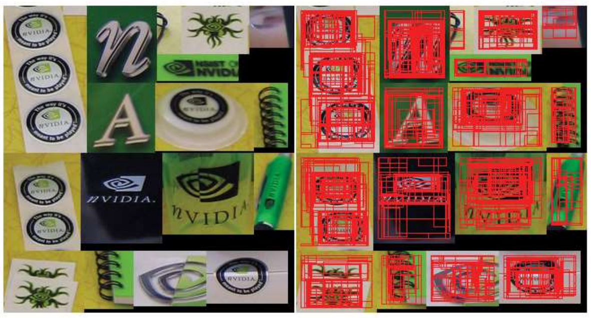

Fig. 4. Connected components of overlapping ROIs packed into a min-area rectangle. On the right, individual bounding boxes are shown. Even small image areas contain many overlapping object proposals.

Since the computational costs mainly depend on the image area, partial magnification of the image will, on average, save computation time compared to magnifying the complete image. (2) Fast R-CNN still offers potential speedups compared to the R-CNN approach – even on small scales. Based on these observations we propose the following modified detection pipeline.

V. Detection Pipeline

As with Fast R-CNN, our network receives an image and a list of ROIs as input. We split the ROIs into two groups – large and small – based on the image area they occupy. As Figure 1 suggests, Fast R-CNN begins to have difficulties to reliably detect structures with a side length smaller than 80 pixels. We therefore choose a side length of 80 pixels as threshold to separate the ROIs into these groups. Like Fast R-CNN, the convolutional layers compute a feature representation for the complete image while the ROI pooling layer generates a fixed-length feature representation by mapping the coordinates of the original ROIs to the corresponding coordinates on the feature map.

Since we are looking for structures with a side length smaller than 80 pixels, the spatial extent of the feature representation is quite small: When using network architectures like CaffeNet or VGG16, the resolution of the last feature map before the ROI pooling is reduced by a factor of 16 compared to the original image. This reduction factor is used by the ROI pooling layer to project the ROI from image coordinates onto the feature map. The spatial extent of the mapped ROI is then being split into $n \times n$ bins, performing a max-pooling operation for each bin. Because of this conversion, we expect the relevant features to have a spatial extent of at most 5 pixels. Because of this, we choose a 3×3 ROI pooling, which seems like a reasonable compromise between spatial information and the dimensionality of the features for the SVM.

For this purpose, we add a 3×3 ROI pooling layer to the network layout in addition to the 7×7 ROI pooling layer employed by Fast R-CNN for classification. We have also experimented with 1×1 pooling which results in a feature of only 512 dimensions. However, we found the feature representation to be significantly inferior to a 3×3 representation. This means that for every ROI we obtain two feature representations of different dimensionality.

All features obtained from the 7×7 ROI-Pooling are directly classified using the standard Fast R-CNN pipeline. For small ROIs, we use the additional 3×3 feature representation as input for an SVM in order to predict a score, which in the following we call the 'objectness' of the ROI – not to be confused with the generic objectness measure developed by [12]. Similarly to [12] a high 'objectness' score indicates a high probability of the ROI containing an object of interest.

ROIs which are classified as positive by the SVM tend to overlap and form clusters in the original image. The clusters of ROIs are fed into a rectangle packing algorithm. From this rectangle packing, a new image is compiled by copying the corresponding image patches from the original image. This packed image is magnified and run through the Fast R-CNN pipeline a second time, computing a new feature representation. The detections from the second pass are mapped back into the original image and combined with the detections from the first pass. Detection results from both passes jointly undergo post-processing, such as bounding box regression and non-maximum suppression.

Using this approach, we are able to avoid upsampling the complete image while processing small object instances on a scale which is suitable for the receptive field of the network and preserving much of the speed of the Fast R-CNN approach.

VI. Packing small objects

In order to keep data transfers between host and GPU memory to a minimum we want to pack all image regions deemed interesting by the SVM into a single new image. Since the computational costs of applying the DCNN is mainly dependent on the area of the image, we use a simple rectangle packing algorithm to find a packing which (approximately) minimizes the image area.

This is done in four steps: (1) The ROIs classified as positive by the SVM tend to overlap strongly. We identify connected components of overlapping ROIs. Each component is treated as a single entity in a rectangular image patch to be packed. (2) By tightly cutting out image patches, we risk to loose important image information. For this reason, each image patch is padded. We found 50 pixels to be sufficient. (3) A rectangle packing algorithm is run to identify the minimum area arrangement for the padded image patches. (4) Based on this arrangement, a new image is created by copying the corresponding image patches from the original image.

Finding an exact solution to the minimum area packing problem is NP-hard [13]. However, several efficient heuristics exist which tend to yield good approximations. The rectangle packing algorithm we use is an implementation of the method described by [13].

A quantitative evaluation of the average reduction in image area is given in Figure 5. The average reduction factor f was

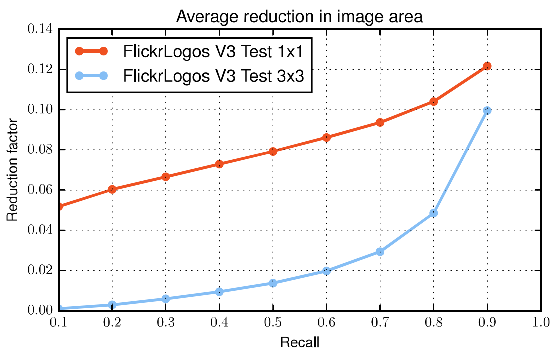

Fig. 5. Average reduction in image area through rectangle packing as a function of the recall. Computational effort is most prominently influenced by image area. Therefore, most of the detection performance can be retained while only requiring a fraction of the computation.

calculated on the *logosonly* subset of the FlickrLogos test set and is given by:

$$f = \frac{1}{N} \sum_{n=1}^{N} \frac{\tilde{A_n}}{A_n} \tag{1}$$

where N is the number of images in the test set, A_n is the area of the n-th image and $\tilde{A_n}$ the area of the packed n-th image. In case of an image in which the SVM does not mark any ROI for closer inspection $\tilde{A_n} = 0$.

An example output of this algorithm when applied to one of the example images in Figure 3 is shown in Figure 4.

VII. Evaluation

In order to evaluate the detection performance on small objects we use the well-known FlickrLogos-32 dataset [3]. This dataset has originally been conceived for logo classification and image retrieval. We supply an evaluation script for object detection as well as new annotations for the object instances resolving previous problems with images containing multiple conflicting logo classes. Additionally, missing annotations have been fixed and new classes have been added. Company logos often consist of both a text and a symbol component. Both components often appear separate from each other. Where applicable, the annotations of product logos have been split into separate classes for the text and the symbol component. The set of images itself has not changed but the division of images into training set and test set has changed to allow for a balanced partition of the newly annotated classes.

For all our experiments, we use a modified [11] version of selective search [7]) as our object proposal algorithm. We filter object proposals with a side length smaller than 15 pixels and place a limit of max. 8000 bounding boxes per image. This rather large number has been chosen since object proposals by selective search are biased towards large objects. Since we are studying the feature representation of small objects we are not primarily concerned about the source or quantity of object proposals. We simply want to make sure that small objects are not already excluded at the proposal stage.

In Table I we compare the results of three experiments: (A) represents our baseline for which we evaluate the default Fast R-CNN approach on FlickrLogos-47 for input images which were not scaled in any way. In (B) we scale up the whole images by a factor of $s = 2.5$ for both training and testing. In (C) we use the original images as input to our modified detection pipeline only magnifying selected image patches. The packed image patches selected by the SVM are upscaled by a factor of $s = 2.5$. The SVM is tuned for a recall of $r = 0.9$ which on average results in packed images which occupy only 10% of their original area. We use the same object proposals for all our experiments.

Table I reports the APs for each individual class, the mAP and the average execution time per image. We can observe that (B) improves the overall detection performance by 4.3 points which represents an increase in performance relative to (A) by 6.4%. Similarly, (C) improves the overall detection performance by 3.4 points which represents an increase in performance relative to (A) by 5.0%.

From a comparison of the individual APs it is evident that nearly all individual classes can potentially benefit from upscaling the images, the only exceptions being the classes 'BMW', 'Google' and 'HP'. This can be partly explained by the fact that 'BMW' and 'Google' are outliers when comparing the average object size between classes. In both classes the average sidelength of an object is unusually large. Even without any magnification the average object in these classes is close to filling the receptive field of the network.

However, the gain in overall detection performance of (B) comes at the cost of greatly increased execution time per image. Relative to (A) the increase in execution time is 245%. Our approach of selectively upscaling ROIs is able to achieve a mAP which is close to the performance of (B) but at a greatly reduced average execution time per image. Relative to (A) the execution only increases by 25%. All timing experiments were conducted on a Quadro K6000 GPU with 12 GB of memory.

In Figure 6 we plot the average object instance size against the change in AP for each class. Unsurprisingly, small objects tend to benefit more strongly from magnification than large objects.

Another interesting observation can be made if we follow the regression lines to the point where objects are so large that no improvements through magnification are to be expected. This point coincides well with our observation from Figure 1 that DCNNs have trouble detection objects which are larger than the receptive field of the network (which in the case of VGG16 is 212 pixels) and further supports our hypothesis that DCNNs work most effectively when the object is scaled in such a way that it approximately fills the receptive field of a neuron on the feature map.

VIII. Conclusion

We have shown that the detection performance of Fast R-CNN depends on the scale of the objects. Larger objects tend to be classified more confidently than small object instances, leading to problems in high-precision applications. We have

TABLE I
PERFORMANCE ON THE FLICKRLOGOS DATASET.

	Adidas symbol	Adidas text	Aldi	Apple	Becks symbol	Becks text	BMW	Carlsberg symbol	Carlsberg text	Chimay symbol	Chimay text	Coca-cola	Corona symbol
A	53.9	24.0	64.4	79.0	70.7	55.0	**79.1**	59.6	52.5	84.2	72.2	56.2	77.0
B	57.2	**41.9**	74.8	84.0	74.5	**58.6**	77.1	61.4	**58.4**	**85.4**	70.0	58.4	**85.2**
C	**61.1**	34.1	**80.9**	**84.9**	**75.6**	57.4	78.6	**63.7**	56.4	84.9	**76.2**	**58.5**	82.7

	Corona text	DHL	Erdinger symbol	Erdinger text	Esso symbol	Esso text	Fedex	Ferrari	Ford	Fosters symbol	Fosters text	Google	Guiness symbol
A	79.0	59.5	92.2	84.8	83.7	13.8	75.5	94.9	81.4	72.8	70.1	**73.7**	75.5
B	**80.4**	**75.0**	**95.7**	**88.4**	84.5	**15.6**	**78.1**	95.0	**85.2**	**78.9**	**73.0**	63.4	78.5
C	78.8	63.1	95.3	82.2	**85.2**	14.0	77.5	**95.4**	82.6	75.3	67.5	71.8	**83.0**

	Guiness text	Hein-eken	HP	Milka	nVidia symbol	nVidia text	Paulaner symbol	Paulaner text	Pepsi symbol	Pepsi text	Ritter-sport	Shell	Singha symbol
A	57.0	45.4	**74.9**	49.2	57.3	19.4	82.8	39.4	35.1	18.2	61.8	80.2	88.8
B	56.7	**68.2**	73.1	**63.0**	61.1	**31.8**	**85.5**	**41.7**	**42.7**	**22.6**	71.4	88.9	**93.4**
C	**58.5**	59.3	71.4	58.8	**67.0**	27.4	85.1	33.9	36.0	16.7	**74.2**	**89.4**	89.9

	Singha text	Star-bucks	Stellart. symbol	Stellart. text	Texaco	Tsingt. symbol	Tsingt. text	UPS	mAP	time
A	81.5	76.3	88.5	88.0	79.9	76.4	70.5	85.3	66.8	**0.82s**
B	72.1	**84.4**	**90.6**	**90.3**	83.9	78.8	73.0	**92.0**	**71.1**	2.83s
C	**81.7**	74.8	88.3	87.7	**89.6**	**82.9**	**73.2**	86.0	70.2	1.03s

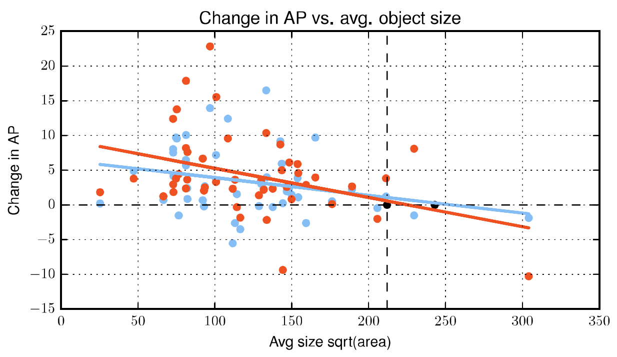

Fig. 6. Average object size in each class vs. the change in AP. Difference between (B) and (A) in red. Difference between (C) and (A) in blue. The improvement is most noticeable for small objects. The regression lines also suggest that the performance gain by upscaling is limited by the receptive field of the DCNN (black dot). This also supports our hypothesis that DCNNs perform best when the objects have approx. the same size as the receptive field.

shown that the low-quality feature representations of small object instances are suitable to predict salient regions which allows us to quickly discard a large fraction of candidate boxes. Small objects can be detected more effectively when they are being magnified. We have shown that treating small objects in this matter does indeed increase the detection performance for very little additional computational resources. Finally, we have shown a link between object size and performance gain. The receptive field of the network is a natural upper bound for upscaling small objects.

ACKNOWLEDGMENT

This work was funded by GfK Verein. The authors would like to thank Carolin Kaiser, Holger Dietrich and Raimund Wildner for the great collaboration. Especially, we would like to express our gratitude for their help in re-annotating the FlickrLogos dataset.

REFERENCES

[1] M. Everingham, L. Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (voc) challenge,” *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2009.
[2] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: common objects in context,” *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312
[3] S. Romberg, L. G. Pueyo, L. R., and R. van Zwol, “Scalable logo recognition in real-world images,” in *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, ser. ICMR ’11. New York, NY, USA: ACM, 2011, pp. 25:1–25:8.
[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, June 2014, pp. 580–587.
[5] A. Krizhevsky, I. Sutskever, and G. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105.
[6] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014.
[7] J. R. R. Uijlings, K. E. A. Sande, T. Gevers, and A. W. M. Smeulders, “Selective search for object recognition,” *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013.
[8] L. Zitnick and P. Dollár, “Edge boxes: Locating object proposals from edges,” in *ECCV*. European Conference on Computer Vision, September 2014.
[9] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” *CoRR*, vol. abs/1506.01497, 2015.
[10] J. Yangqing, E. Shelhamer, D. J., S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” *arXiv preprint arXiv:1408.5093*, 2014.
[11] A. Winschel, C. Eggert, and R. Lienhart, “Diversity in object proposals,” *CoRR*, vol. abs/1603.04308, 2016.
[12] B. Alexe, T. Deselaers, and V. Ferrari, “Measuring the objectness of image windows,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 11, pp. 2189–2202, Nov 2012.
[13] R. E. Korf, “Optimal rectangle packing: Initial results,” in *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS 2003), June 9-13, 2003, Trento, Italy*, 2003, pp. 287–295.