EVALUATING CONTEXT FOR DEEP OBJECT DETECTORS

Osman Semih Kayhan   Jan C. van Gemert

Computer Vision Lab, Delft University of Technology

ABSTRACT

Which object detector is suitable for your context sensitive task? Deep object detectors exploit scene context for recognition differently. In this paper, we group object detectors into 3 categories in terms of context use: no context by cropping the input (RCNN), partial context by cropping the featuremap (two-stage methods) and full context without any cropping (single-stage methods). We systematically evaluate the effect of context for each deep detector category. We create a fully controlled dataset for varying context and investigate the context for deep detectors. We also evaluate gradually removing the background context and the foreground object on MS COCO. We demonstrate that single-stage and two-stage object detectors can and will use the context by virtue of their large receptive field. Thus, choosing the best object detector may depend on the application context.  

1. INTRODUCTION

Objects are rarely photographed alone. An image may contain several other objects or varying scene background. This background context may correlate with the object and thus possibly exploited by a learned object detector such as detecting a chair next to a table and missing the same chair in a football field. For some applications, the context should not matter like an object placed in a box of a retail system. For other applications, the context is an important cue, such as images from an MRI scanner. In this paper we evaluate the effect of context on popular deep object detectors.

For deep object detectors, there are works on the context around the object [1], as scene context [2, 3, 4], as combinations of local parts and the global image structure [5, 6], by using multi-scale fusion and attention [7], and by using recurrent neural networks [8]. Besides, to have a robust decision, [9] uses mixup augmentation and [10] investigates how to disentangle object from its co-occurring context. [11] investigates the contextual effect on visual recognition with various ways and compare with human performance. [12] explores the bounds of object classes by using contextual information and show the cases when is beneficial to use or discard. [13] indicates that contextual information can make the detectors hallucinate some objects and their locations whilst the objects are not even in the images. In addition, [14, 15] show absolute spatial position of object can affect the performance. Differently, we classify object detectors in terms of context use and analyze the effect of context for these detector types.

Popular deep object detectors follow either a single-stage or a two-stage approach. Two-stage detectors consist of class-agnostic region proposals and detection parts. RCNN [16] is the earliest deep two-stage detector that crops the ROIs from the input image before feeding the CNN backbone, without accessing context. Faster RCNN [17] introduces the trainable Region Proposal Network (RPN) and each candidate region is cropped from deep featuremap. Faster RCNN is the most common two-stage detector and a great inspiration to other detectors [5, 6, 18, 19], hence we evaluate Faster-RCNN as the prototypical two-stage detector. Single-stage detectors do not use any proposal method and obtain the detection in a single run. YOLO [20] treats detection as a regression problem and detects the objects from a full image. In YOLOv2 [21] and YOLOv3 [22], the method is improved by using a deeper backbone model, multi-scale training, high resolution input and anchor boxes. SSD [23] predicts category scores and box offsets for a fixed set of anchors from different scales. RetinaNet [24] proposes focal loss which focuses on the hard training samples and converges faster. Effi-
cientDet [25] scales the model and proposes fast multi-scale feature fusion. For our experiments, we choose YOLOv3 since it uses anchor boxes and multi-scale training, thus comparable with Faster RCNN.

In this paper, we assume that context is formed as everything around the object including other objects. We classify object detectors on how they use context (Fig. 1): (i) no context (RCNN), (ii) partial context (Faster RCNN) and (iii) full context (YOLO) on a fully-controlled dataset, see Fig. 2. Using context is beneficial if the object correlates with its environment. However, the performance is reduced when the context is incoherent. We demonstrate the effect of context on detector performance by increasing the context around to object in each testing case. Also, we indicate how much contribution the detector can obtain by only using the context information without a visible main object.

We have the following contributions: First, we show that modern deep object detectors can access context by virtue of their receptive field size even if the object regions are cropped from the feature map. Second, the effect of context is evaluated quantitatively on most common object detection networks. To conclude, we indicate that single and two stages networks employ contextual information except methods crop the ROIs from the input such as RCNN.

2. EXPERIMENTAL EVALUATION OF CONTEXT

We analyze the effects of various contextual correlations for common detector types on a fully-controlled context dataset and evaluate context with natural images.

We categorize deep object detectors (see Fig. 1) as:

**Crop-Input.** We base this class of detectors on the seminal RCNN [16] approach, which originally uses class-agnostic object proposal bounding boxes [26] which are cropped from the input and fed to a CNN backbone for feature extraction, then the extracted features are used for detection. Since the method crops the proposals from the input image before feature extraction, it does not access any context beyond the bounding box, thus we call it 'no context' method. In reality, a network may retrieve minimal context between an object and the area inside the bounding box.

**Crop-FM.** This class of detectors crops bounding boxes from CNN feature maps. The seminal example is based on Faster RCNN [17] which has two stages: a detection head for object classification and an RPN which outputs candidate object boxes. These boxes are cropped by ROI pooling from feature maps. These feature maps are deep in the network and thus are the result of convolutions with a large receptive field. Due to such large receptive fields, the feature map crops include context information beyond the cropped regions which can be exploited for recognition.

**NoCrop.** This class of object detectors does not crop at all and includes most of the single-stage object detectors such as YOLO detectors [20, 21, 22]. Predictions are made by using the full feature map and thus can exploit all context.

2.1. Evaluating object-context correlations

It is difficult to vary the correlation between an object and its context in real images. Thus, we create a fully controlled context-sensitive dataset from the 10-class Fashion MNIST [27]. To vary object-context correlation we create 6,000 images (2000 per training, validating and test set) to form the Quadrant-FMNIST (Q-FMNIST) dataset by placing images in quadrants. We create a 2-class object detection problem where the top-left quadrant has the object of interest (class-1: ‘Pullover’ and class-2: ‘Shirt’) and the other 3 quadrants are filled with images of other 8 classes which is how we vary object-context correlation. Namely, these 8 classes become background for each image. We have 5 degrees of object-context correlations, shown in Fig. 2.

- **No context:** The background is full black.
- **Uncorrelated context:** The 3 locations of the template are filled randomly from the 8 context classes.
- **Semi-correlated context:** Up-right corner, bottom-left corner and bottom-right corner locations of class-1 are filled with respectively ‘Dress’, ‘Coat’ and ‘Bag’ classes (common context for class-1) by the probability of 75% and with ‘Trouser’, ‘Sandal’ and ‘Ankle bot’ classes by 25%. Similarly, class-2 images are filled with ‘Trouser’, ‘Sandal’ and ‘Ankle bot’ classes (common context for class-2) by 75% and ‘Dress’, ‘Coat’ and ‘Bag’ classes by 25%.
- **Fully-correlated context:** This mode represents the cases when the objects are observed always in similar context, such as cows and horses are together on a grass field. The context is placed in a structured way in terms of location, the same context occurs in the same place in the train and test set. For class-1 images, coat, dress and bag objects are placed respectively up-right corner, bottom-left corner and bottom-right corner.
right corner. Likewise, for class-2, sandal, trouser and ankle bot objects are placed with the same order as the coat, dress and bag objects. The context objects are not alternated.

**Anti-correlated context:** Training set is built by using fully-structured context train set, however, testing is done by filling with incoherent context by switching the class-specific context. This mode illustrates the cases when the object is seen in an unusual context, such as a shoe in a plate. Class-1 images are filled with 'Trouser', 'Sandal' and 'Ankle bot' classes and likewise class-2 images are filled with 'Dress', 'Coat' and 'Bag' classes respectively up-right corner, bottom-left corner and bottom-right corner. Namely, the context of class-1 and class-2 are swapped.

We instantiate each of the three deep object detector classes in Fig. 1 with two convolution layers with 6 and 16 3x3 filters, two max pooling layers, one fully connected layer with 128 neurons and softmax classifier. We use the ground truth location to crop bounding boxes for the Crop-Input model. For the Crop-FM model, RoiAlign [28] is used for cropping ROIs from the featuremap. Each method is trained 5 times for 15 epochs with the AdaDelta optimizer.

**Results.** In Table 1, the Crop-Input detector disregards all context. For the Crop-FM and NoCrop detectors, the 'no context’ setting gives an object-only baseline. Adding 'semi-structured context' improves, and adding 'fully correlated context' even more. Interestingly, adding 'uncorrelated context' decreases results, whereas 'anti-correlated context' completely misclassifies the objects. As the NoCrop detector uses the full image, it is more sensitive to context changes than the Crop-FM. Being more sensitive can be an advantage for correlated object-contexts, yet can be detrimental for random context or when an object is placed outside the usual context where the context may outweigh the object itself.

### 2.2. Evaluating context on natural images

We investigate the context for natural images on the COCO minival 2014 split [29]. We evaluate two settings: **Hiding background** and **Hiding foreground**, see Fig. 3. We begin with the ground truth object bounding box. For hiding BG, we start without any context using a black background and incrementally add more background pixels on each side of the object. For hiding FG, we start without an object and make the bounding box black and incrementally add object pixels towards the center on each side. We increase pixels in the range ∈ {0, 5, 10, 25, 50, 100, 150, 200, 250, ...} until reaching the full image for both settings. If the context of an image reached an image border on one side, it stops there, yet, the change continues on the other object sides.

For Crop-Input we use a variation of R-CNN [16] where we use a softmax classifier based on an ImageNet pretrained Alexnet. The object crops are resized as 227x227 and trained from scratch 35 epochs with SGD for an initial learning rate of 1e-3. For Crop-FM, we use Faster RCNN with ROI Align to crop ROIs from the featuremap. An existing COCO pretrained network is used with a Resnet-50 backbone with an FPN [18]. For NoCrop, we use YOLO version 3 [22] which is fully-convolutional and has 75 layers with skip connections. To evaluate hiding BG and FG setups, we use classification accuracy. The effect of localization is minimized as following: For RCNN, ground truth box locations are used to crop the objects from the input image. For Faster RCNN, IoU threshold is set as 0.25. For YOLO, the prediction is counted as correct if the center location of predicted and ground truth boxes for correct class label remain in the same grid cell.

**Hiding BG.** Results in Fig. 4 (left) confirms that the Crop-Input R-CNN does not depend on context. The Crop-FM Faster RCNN outperforms the NoCrop YOLO when no context is available. Both detectors have their peak when 100 pixels is added to each side of the object: 90.2% for Faster RCNN and 87% for YOLO. We hypothesize that the object-context correlation is best around 100 pixels and decreases as more background is taken into account.

For **Hiding BG**, in Fig. 5 (top) we show the classes that have the smallest and the largest difference between no context and full context. For Faster RCNN, bottle class is highly context dependent and loses 61% without context. Besides, sink performs 2% better without context. For YOLO, baseball glove, handbag and book lose more than 65% due to lack of context. However, cat, bed and giraffe obtain better result when no context is used. Interestingly, 3/4 of classes obtains better result with using some amount of context rather than using full context. The fact also explains the performance increases in Fig. 4 (left) between 50 and 300 context ranges.

### Table 1: Accuracy on Q-FMNIST. Context affects Crop-FM and NoCrop detectors. For correlated context results improve. Uncorrelated context is worse than no context. Anti-correlated context is detrimental.

| Context       | Crop-Input | Crop-FM  | NoCrop  |
|---------------|------------|----------|---------|
| No context    | 87.9 ± 1.0 | 88.1 ± 0.4 | 86.9 ± 0.7 |
| Uncorrelated  | 87.9 ± 1.0 | 86.7 ± 0.7 | 83.7 ± 1.1 |
| Semi-correlated | 87.9 ± 1.0 | 89.3 ± 0.8 | 90.4 ± 0.5 |
| Fully-correlated | 87.9 ± 1.0 | 90.7 ± 0.1 | 100 ± 0.0 |
| Anti-correlated | 87.9 ± 1.0 | 1.8 ± 0.2  | 0.0 ± 0.0 |

![Fig. 3: Example of hiding background (left) and hiding foreground (right). Hiding BG incrementally adds background pixels. Hiding FG incrementally adds object pixels.](image-url)
Fig. 4: Hiding Background and Foreground experiments on 80-class COCO with RCNN, Faster RCNN and YOLO. The x-axis shows how many pixels of context (Hiding BG) or object (Hiding FG) is available from each side. Both plots confirm Faster RCNN is less sensitive to context than YOLO. For Hiding BG, 50-100 pixels extra has best object-context correlation, which reduces when adding more background. For Hiding FG, R-CNN needs only 100 pixels to classify an object.

Fig. 5: Evaluation of class-specific result of Faster RCNN and YOLO on COCO. Data points represent the 3 best and 3 worst-performing classes for each method according to difference between context/object and full image.

**Hiding FG.** Results in Fig. 4 (right) shows that YOLO is best in exploiting context when the full object is removed. Interestingly, the performance of Faster RCNN is still far from random for 80 classes without actually seeing the object while RCNN scores truly random with $\frac{1}{80} \approx 1.3\%$. RCNN can clas-
4. REFERENCES

[1] S. Gidaris and N. Komodakis, “Object detection via a multi-region and semantic segmentation-aware cnn model,” in ICCV, 2015.

[2] S. Gupta, B. Hariharan, and J. Malik, “Exploring person context and local scene context for object detection,” arXiv preprint arXiv:1511.08177, 2015.

[3] J. Sun and D. W. Jacobs, “Seeing what is not there: Learning context to determine where objects are missing,” in CVPR, 2017.

[4] Y. Liu, R. Wang, S. S., and X. Chen, “Structure inference net: Object detection using scene-level context and instance-level relationships,” in CVPR, 2018.

[5] Y. Zhu, C. Zhao, J. Wang, X. Zhao, Y. Wu, and H. Lu, “Couplenet: Coupling global structure with local parts for object detection,” in ICCV, 2017.

[6] X. Fan, H. Guo, K. Zheng, W. Feng, and S. Wang, “Object detection with mask-based feature encoding,” arXiv preprint arXiv:1802.03934, 2018.

[7] J.-S. Lim, M. Astrid, H.-J. Yoon, and S.-I. Lee, “Small object detection using context and attention,” arXiv preprint arXiv:1912.06319, 2019.

[8] S. Bell, C Lawrence Zitnick, K. Bala, and R. Girshick, “Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks,” in CVPR, 2016.

[9] Z. Zhang, T. He, H. Zhang, Z. Zhang, J. Xie, and M. Li, “Bag of freebies for training object detection neural networks,” arXiv preprint arXiv:1902.04103, 2019.

[10] K. K. Singh, D. Mahajan, K. Grauman, Y. J. Lee, M. Feiszli, and D. Ghadiyaram, “Don’t judge an object by its context: Learning to overcome contextual bias,” in CVPR, 2020.

[11] M. Zhang, C. Tseng, and G. Kreiman, “Putting visual object recognition in context,” arXiv preprint arXiv:1911.07349, 2019.

[12] E. Barnea and O. Ben-Shahar, “Exploring the bounds of the utility of context for object detection,” in CVPR, 2019.

[13] O. S. Kayhan, B. Vredeburg, and J. C. van Gemert, “HALLUCINATION in object detection — a study in visual part verification,” in ICIP, 2021.

[14] O. S. Kayhan and J. C. van Gemert, “On translation invariance in CNNs: Convolutional layers can exploit absolute spatial location,” in CVPR, 2020.

[15] M. Manfredi and Y. Wang, “Shift equivariance in object detection,” in ECCV workshop, 2020.

[16] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Region-based convolutional networks for accurate object detection and segmentation,” PAMI, 2015.

[17] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in Advances in neural information processing systems, 2015, pp. 91–99.

[18] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in CVPR, 2017.

[19] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, “Deformable convolutional networks,” in ICCV, 2017.

[20] J. Redmon and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.

[21] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” in CVPR, 2017.

[22] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” arXiv preprint arXiv:1804.02767, 2018.

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in ECCV, 2016.

[24] T.-Y. Lin, P. Goyal, R. Girshick, and A. Farhadi, “Focal loss for dense object detection,” in ICCV, 2017.

[25] M. Tan, R. Pang, and Q. V. Le, “Efficientdet: Scalable and efficient object detection,” arXiv preprint arXiv:1911.09070, 2019.

[26] J. R. R. Uijlings, K. E. A. van de 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.

[27] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” arXiv preprint arXiv:1708.07747, 2017.

[28] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in ICCV, 2017.

[29] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in ECCV, 2014.