Toward Scale-Invariance and Position-Sensitive Region Proposal Networks

Hsueh-Fu Lu, Xiaofei Du(✉), and Ping-Lin Chang

Umbo Computer Vision, London, UK
{topper.lu,xiaofei.du,ping-lin.chang}@umbocv.com,
https://umbocv.ai

Abstract. Accurately localising object proposals is an important pre-condition for high detection rate for the state-of-the-art object detection frameworks. The accuracy of an object detection method has been shown highly related to the average recall (AR) of the proposals. In this work, we propose an advanced object proposal network in favour of translation-invariance for objectness classification, translation-variance for bounding box regression, large effective receptive fields for capturing global context and scale-invariance for dealing with a range of object sizes from extremely small to large. The design of the network architecture aims to be simple while being effective and with real-time performance. Without bells and whistles the proposed object proposal network significantly improves the AR at 1,000 proposals by 35% and 45% on PASCAL VOC and COCO dataset respectively and has a fast inference time of 44.8 ms for input image size of 640^2. Empirical studies have also shown that the proposed method is class-agnostic to be generalised for general object proposal.

Keywords: Object detection · Region proposal networks
Position-sensitive anchors

1 Introduction

Object detection has been a challenging task in computer vision [6,17]. Significant progress has been achieved in the last decade from traditional sliding-window paradigms [7,28] to recent top-performance proposal-based [27] detection frameworks [9–11]. A proposal algorithm plays a crucial role in an object detection pipeline. On one hand, it speeds up the detection process by considerably reducing the search space for image regions to be subsequently classified. On the other hand, the average recall (AR) of the object proposal method has

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-01237-3_11) contains supplementary material, which is available to authorized users.

© Springer Nature Switzerland AG 2018

been shown notably correlating with the precision of final detection, in which AR essentially reveals how accurate the detected bounding boxes are localised comparing with the ground truth [13].

Instead of using low-level image features to heuristically generate the proposals [27,30], trendy methods extract high-level features by using deep convolutional neural networks (ConvNets) [12,26,29] to train a class-agnostic classifier with a large number of annotated objects [15,21,23]. For general objectness detection, such supervised learning approaches make an important assumption that given enough number of different object categories, an objectness classifier can be sufficiently generalised to unseen categories. It has been shown that learning-based methods indeed tend to be unbiased to the dataset categories and learn the union of features in the annotated object regions [4,13,15,20]. Despite their good performance [5,12,23], there is still much room to improve the recall especially for small objects and accuracy for the bounding box localisation [2,14,16,20].

To tackle object detection using ConvNets at various scales and for more accurate localisation, prior works adopted an encoder-decoder architecture with skip-connections [24] for exploiting low-resolution strong semantic and high-resolution weak semantic features [16], used position sensitive score maps for enhancing translation variance and invariance respectively for localisation and classification [5], and used a global convolutional network (GCN) component for enlarging valid receptive field (VRF) particularly for capturing larger image context [19].

In this paper, we devise an advanced object proposal network which is capable of handling a large range of object scales and accurately localising proposed bounding boxes. The proposed network architecture embraces fully convolutional networks (FCNs) [18] without using fully-connected and pooling layers to preserve spatial information as much as possible. The design takes simplicity into account, in which the features extracted by ConvNets are entirely shared with a light-weight network head as shown in Fig. 2.

Ablation studies have been conducted to show the effectiveness of each designed component. We have empirically found that GCN and position-sensitivity structure can each individually improves the AR at 1,000 proposals. As shown in Tables 2 and 3, evaluating the baseline model on PASCAL VOC and COCO dataset, GCN brings performance gains from 0.48 and 0.42 to 0.59 (22%) and to 0.54 (29%) respectively, and, the use of position-sensitivity to 0.61 (26%) and to 0.45 (6%) respectively. Using them together can furthermore boost the scores to 0.65 (35%) and to 0.61 (44%) respectively. Together the proposed framework achieves the state of the art and has a real-time performance.

2 Related Works

Traditional object proposal methods take low-level image features to heuristically propose regions containing objectness. Methods such as Selective Search [27], CPMC [3] and MCG [1] adopt grouping of multiple hierarchical segmentations to produce the final proposals. Edge boxes [30] on the other hand takes an assumption that objectness is supposed to have clearer contours. Hosang et al. [13] have comprehensively evaluated different proposal methods. Learning-based proposal approaches have gained more attentions recently. DeepBox [15] uses convolutional neural network to re-rank object proposals based on other bottom-up non-learning proposal methods. Faster R-CNN [23] trains a region proposal network (RPN) on a large number of annotated ground truth bounding boxes to obtain high-quality box proposals for object detection. Region-based fully-convolutional network (R-FCN) [5] introduces position-sensitive score maps to improve localisation of the bounding boxes at detection stage. Feature pyramid network (FPN) [16] takes multi-scale feature maps into account to exploit scale-invariant features for both object proposal and detection stages. Instead of using feature pyramids and learning from ground truth bounding boxes, DeepMask [20] and SharpMask [21] use feed-forward ConvNets trained by ground truth masks and exploit multi-scale input images to perform mask proposals to achieve state-of-the-art performances.

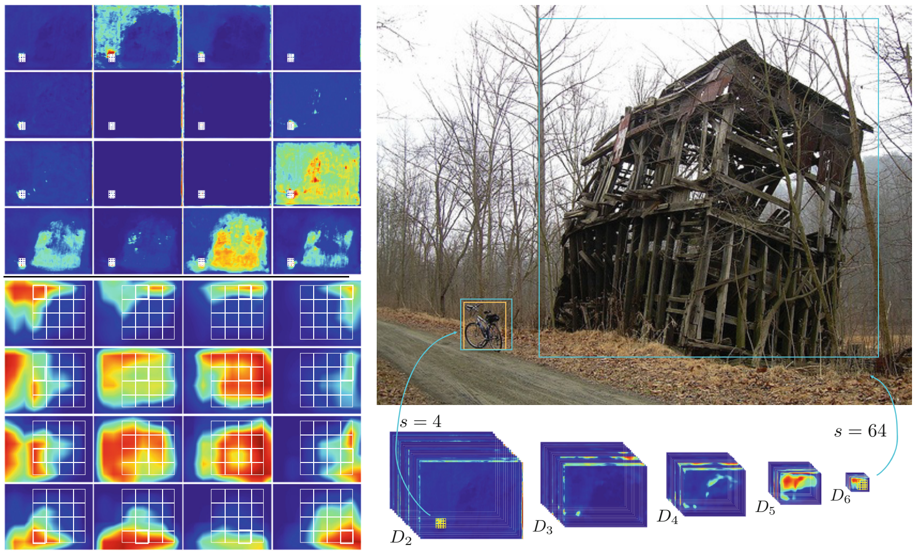

Fig. 1. The proposed method. **Left:** The position-sensitive score maps and windows with k^2 grids ($k = 4$) in yellow at D_2 and D_6 shown at the top and bottom respectively. One can see the D_2 activates on small objects while D_6 activates on extreme large ones. Note that D_6 is enlarged for visualisation, in which the window size is in fact identical to the one shown in D_2. **Right:** The windows are mapped into anchors in cyan in the input image with sizes of the multiple of layer stride s. **Both:** The bounding box in orange is the only labeled ground truth (in `bike` category) on this image from PASCAL VOC 2007. The large object on the right has no ground truth but the proposed class-agnostic method can still be generalised to extract its objectness features as shown in the visualised D_6 feature maps.

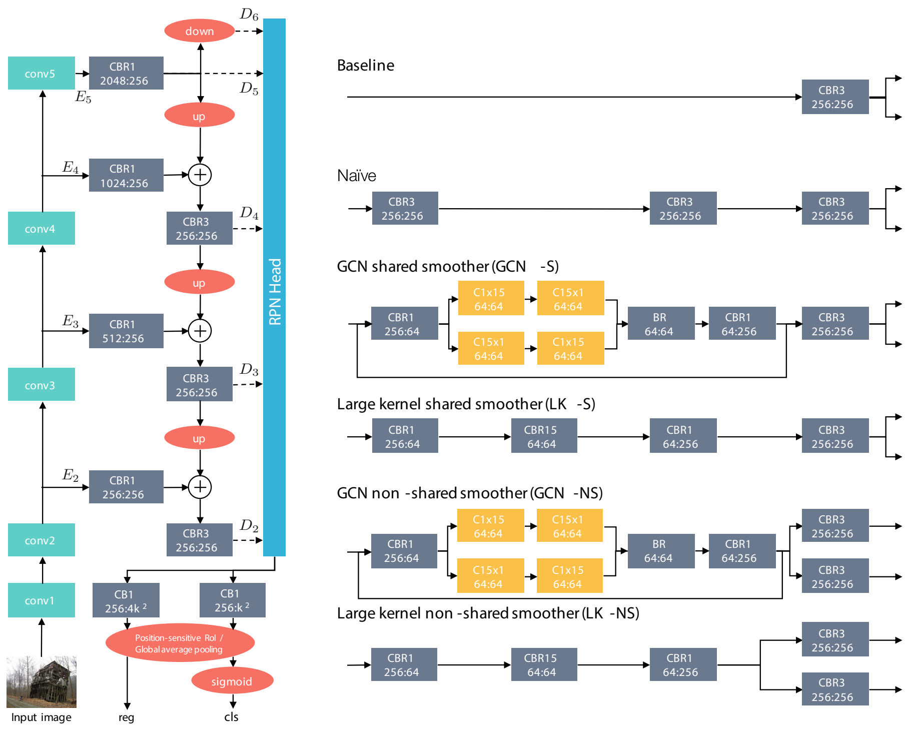

Fig. 2. The overall proposed system architecture. **Left:** The ResNet together with the feature pyramid structure form the general backbone of RPN heads. **Right:** The structures of different RPN heads. **Both:** Rectangles are components with learnable parameters to train and ellipses are parameter-free operations. Dash arrow indicates that the RPN head is shared by all feature pyramid levels.

3 Proposed Method

Inspired by FPN [16] and R-FCN [5], the proposed object proposal method is devised in favour of scale-invariance and position-sensitivity to retain both invariance and variance on translation for respectively classifying and localising objects. We also take VRF into account to learn objectness from a larger image spatial context [19]. In addition, instead of regressing and classifying a set of anchors using default profile (i.e., scale and aspect ratio) by a fixed (3×3) convolutional kernel in certain layers [16,23], we propose directly mapping anchors from sliding windows in each decoding layer together with sharing the position-sensitive score maps. The overall ConvNets takes an input image with arbitrary size to bottom-up encode and top-down decode features with skip connections to preserve object locality [24]. Scale-invariance as one of the important traits of the proposed method is thus achieved by extracting multi-scale features from the input image. These semantically weak to strong features are then feed into a series of decoding layers being shared by a RPN head. Anchors are generated by a dense sliding window fashion shared by a bank of position sensitive score maps. In the end, the network regresses the anchors to localise objects (*reg* for short) and classifies the objectness with scores (*cls* for short).

3.1 Encoder

The encoder is a feed-forward ConvNet as the backbone feature extractor, which scales down by a factor of 2 several times. Although the proposed method can be equipped with any popular ConvNet architectures [26,29] for the backbone, ResNet [12] is adopted particularly for its FCN structure being able to retain the local information as much as possible. ResNets are structured with residual *blocks* each consisting of a subset of ConvNets. We note the conv2, conv3, conv4 and conv5 blocks from the original paper [12] as $\{E_2, E_3, E_4, E_5\}$ with the corresponding dense sliding window strides $s = \{4, 8, 16, 32\}$ in regard to the input image.

3.2 Decoder

The decoder recovers the feature resolution for the strongest semantic feature maps from low to high with skip connections in between the corresponding encoder and decoder layers. The skip connection is substantial for the accuracy of bounding box proposal as it propagates detail-preserving and position-accurate features from the encoding process to the decoded features which are later shared by the RPN head.

Specifically with ResNet, the decoding process starts from $E5$ using 1×1 convolution and 256 output channels for feature selection followed by batch normalisation (BN) and rectified linear unit (ReLU) layers, which together we brief as CBR$\{\cdot\}$ where $\cdot$ is the kernel size. Likewise, each skip connection at a layer takes a CBR1 with 256 output channels. The bottom-up decoding process is therefore done by using bilinear upsampling followed by element-wise addition with the CBR1 selected features from the encoding layers. A CBR3 block is inserted in each decoding layer right after the addition for the purpose of de-aliasing. We note the decoding layers as $\{D_2, D_3, D_4, D_5\}$ corresponding to $\{E_2, E_3, E_4, E_5\}$ in the encoding layers. An extra D_6 is added by directly down sampling D_5 for gaining an even larger stride $s = 64$, which is in favor of extremely large objectness detection.

3.3 RPN Heads

A RPN head is in charge of learning features across a range of scales for *reg* and *cls*. The learnable parameters in the RPN head share all features in the decoding layers to capture different levels of semantics for various object sizes. We will show that the design of a RPN head has a significant impact on the final proposal accuracy in Sect. 4. We show a number of different RPN head designs in Fig. 2. Each head takes 256 channel feature map as input and outputs two sibling CB1 blocks for *reg* and *cls* with $4 \times k^2$ and k^2 channels respectively, where k^2 is the number of regular grids for position-sensitive score maps described in Sect. 3.4. We regard the state-of-the-art RPN used in FPN [16] as a *Baseline* method, in which a CBR3 block is adopted for fusing multi-scale features. Our *Baseline* implementation, which is a bit different from [16], uses BN and ReLU which have been found helpful in converging the end-to-end training.

Inspired by GCN within residual structure [19], we hypothesise that enlarging VRF to learn from larger image context can improve the overall object proposal performance. For the *GCN shared smoother* (GCN-S) and *Large kernel shared smoother* (LK-S), a larger convolution kernel (15×15) is inserted before the CBR3 smoothing. Additionally their non-shared smoother counterpart (GCN-NS and LK-NS) are also compared.

To study the effect of model capacity and the increased number of parameters, a *Naïve* head is taken into account, which is simply added with more CBR3 blocks to approximately match the number of learnable parameters compared with other RPN heads. Table 1 lists the number of parameters of all RPN heads. Compared with the *Baseline*, the numbers of parameter ratio of the other models are within a 0.015 standard deviation.

3.4 Position-Sensitive Anchors

We argue that using a default set of scales and aspect ratios to map anchors from a constant-size convolution kernel can potentially undermine the accuracy of *reg* and *cls*. This could be due to the mismatch of the receptive field of network and the mapped anchors. Prior works have used such strategy [5,16,23] with little exploration of other varieties. To improve the fidelity of relationship between features and anchors with respect to the receptive field of ConvNets, in the proposed method, at each layer, the size of an anchor is calculated by $(w \cdot s) \times (h \cdot s)$ where w and h are the width and height of the sliding window.

Since the anchor and the sliding window are now naturally mapped, position-sensitive score maps can be further exploited for improving the accuracy of localisation. Fig. 1 illustrates the stack of score maps for k^2 regular grids in the sliding window. Each grid in the window takes average of its coverage on the corresponding score map (i.e., average pooling). All k^2 grids then undergo a global average pooling to output 4-channel t and 1-channel o for the final *reg* and *cls* result respectively. We further feed o to an activation function *sigmoid* for evaluating the objectness score. Details of position-sensitive pooling can be found in [5]. In this paper we use $k = 4$ for illustration as well as for all experiments.

3.5 Implementation Details

In this paper, all the experiments were conducted with ResNet-50 with the removal of average pooling, fully-connected and softmax layers in the end of the original model. We do not use conv1 in the pyramid due to the high memory footage and too low-level features which contribute very little for semantically representing objectness. The architecture is illustrated in Fig. 2. A set of window sizes $w : h = \{8 : 8, 4 : 8, 8 : 4, 3 : 9, 9 : 3\}$ are used for the dense sliding windows at each layer for generating anchors. At the most top D_6 and bottom D_2 layer, additional window sizes $\{12 : 12, 6 : 12, 12 : 6, 12 : 4, 4 : 12\}$ and $\{4 : 4, 2 : 4, 4 : 2\}$ are respectively used for discovering extremely large and small objectness.

The proposed position-sensitive anchors mapped from the windows are all inside the input image, but the bounding boxes regressed from anchors can possibly exceed the image boundary. We simply discard those bounding boxes exceeding the image boundary. In addition, the number of total anchors depends on the size of input image and the used anchor profile. The effect of anchor number is discussed in the supplementary material.

Training. In each image, a large amount of anchors are generated across all decoding layers to be further assigned positive and negative labels. An anchor having intersection-over-union (IoU) with any ground truth bounding box greater than 0.7 is assigned a positive label p and less than 0.3 a negative label n. For each ground truth bounding box, the anchor with the highest IoU is also assigned to a positive label, only if the IoU is greater than 0.3. This policy is similar to [23] but with the additional lower bound for avoiding distraction of outliers. N_A anchors (half positive and half negative anchors) are selected for each training iteration. The model can be trained end-to-end with N_B mini-batch images together with the sampled anchors using a defined loss:

$$L = \frac{1}{N_B \cdot N_A} \sum_{i=1}^{N_B} \left[\sum_{j=1}^{N_A} \left[L_{reg}(t_{i,j}^{p}, t_{i,j}^{*}) + L_{cls}(o_{i,j}^{p}) \right] + \sum_{j=1}^{N_A} L_{cls}(o_{i,j}^{n}) \right], \quad (1)$$

where t is the regressed bounding box with t^* as its ground truth correspondent, and o is the objectness score. L_{reg} is the smooth L_1 loss taking the difference between normalised bounding box coordinates with the ground truth as defined in [9], and L_{cls} the cross-entropy loss. We use stochastic gradient descent (SGD) with momentum of 0.9, weight decay of 10^{-4} and exponential decay learning rate $l_e = l_0 b^{-\lambda e}$, in which the e is the epoch number and we set $l_0 = 0.1$ and $\lambda = 0.1$ for the base $b = 10$.

4 Empirical Studies

We have conducted comprehensive experiments for comparing different RPN heads as well as ablation studies to show the impact of position-sensitive score maps. The experiment platform is equipped with an Intel(R) Xeon(R) CPU E5-2650 v4@2.20GHz CPU and Nvidia Titan X (Pascal) GPUs with 12 GB memory. Such hardware spec allowed us to train the models with batch size N_B listed in Table 1. Note that we particularly focus on conducting ablation studies on different components. In all experiments we therefore did not exploit additional tricks for boosting the performance such as using multi-scale input images for training [11] and testing [12], iterative regression [8], hard example mining [25], etc.

Table 1. The number of parameters in the different models and the corresponding inference time T in *ms* averaged on the number of all testing images

	w/o position-sensitive				w/ position-sensitive			
	# params	N_B	$\mathrm{T}_{\texttt{07test}}$	$\mathrm{T}_{\texttt{minival}}$	# params	N_B	$\mathrm{T}_{\texttt{07test}}$	$\mathrm{T}_{\texttt{minival}}$
Baseline	26,858,334	28	26.6	58.2	26,875,104	18	35.7	79.5
Naïve	28,039,006	18	32.2	-	28,055,776	14	41.5	-
GCN-S	27,137,630	18	34.3	76.3	27,154,400	14	44.1	96.1
LK-S	27,81,3470	18	45.2	-	27,830,240	14	55.1	-
GCN-NS	27,727,966	16	35.9	83.5	27,744,736	12	44.8	103.6
LK-NS	28,403,806	16	48.8	-	28,420,576	12	57.5	-

Baseline Model. The implementation of our *Baseline* model, with or without using position-sensitive score maps, differ from the original FPN [16] in the use of BN and ReLU in the RPN head, as well as the de-aliasing CBR3 block in each layer. In addition, the evaluation in their paper was conducted with rescaling image short side to 800 pixels. The rest of setting such as anchor generation, the number of pyramid layers, etc. are remained the same. Note that such discrepancy do not affect the ablation studies here to compare the baseline architecture. The main purpose is to assess performance gains when adding other network components.

Evaluation Protocol. All models are evaluated on PASCAL VOC [6] and COCO [17]. For Pascal VOC we used all train and validation dataset in 2007 and 2012 (denoted as `07+12`), which has in total 16,551 images with 40,058 objects, and report test result on the 2007 test dataset (denoted as `07test`) consisting of 4,952 images with 12,032 objects. For COCO we employed the union of train and a subset of validation set for in total 109,172 images and 654,212 objects (denoted as `trainval35k`), and report test results on the rest of validation set for 4,589 images and 27,436 objects (denoted as `minival`). Our evaluation is consistent with the official COCO evaluation protocol, in which areas marked as "crowds" are ignored and do not affect detector's scores [17].

In order to perform mini-batch training, we rescaled images in `07+12` with the long side fixed and zero-pad along the rescaled short side to 640 $\times$ 640 for batching, and for images in `trainval35k`, the short side were fixed to 768 with random crop along the rescaled long side to 768 $\times$ 768. For testing, images in `07test` and `minival` are padded to have width and height being the closest multiple of the maximum stride (i.e., $s = 64$), to avoid rounding errors. All models were trained for 40 epochs which roughly take 30 and 90 h for PASCAL VOC 07+12 and COCO trainval35k respectively.

Following [4,13,17], we evaluated models at AR with different proposal numbers of 10, 100 and 1,000 $\{AR^{10}, AR^{100}, AR^{1k}\}$ and the area under the curve (AUC) of recall across all proposal numbers. Besides, we also evaluated models at AR for different object area a: small ($a < 32^2$), medium ($32^2 < a < 96^2$) and large ($a > 96^2$) with 1,000 proposals $\{AR_s^{1k}, AR_m^{1k}, AR_l^{1k}\}$. It is worth noting that COCO has more complex scenes with diverse and many more small objects than PASCAL VOC does [17,22]. We therefore evaluated all models with PASCAL VOC while selected the top-performance GCN-S and GCN-NS models for COCO evaluation. In the tables, numbers with underline indicate the highest score of a metric among models with different RPN heads, and numbers in bold indicate the highest score of a metric among models with or without using position-sensitivity.

4.1 Impact of Using GCN

The results of Tables 2 and 3 reveal that by adding GCN in the RPN head, the overall AR can be remarkably improved regardless the number of considered proposals or if the position-sensitivity is employed. This can be also observed in Fig. 3 in which GCN-S and GCN-NS curves are always on the top of others.

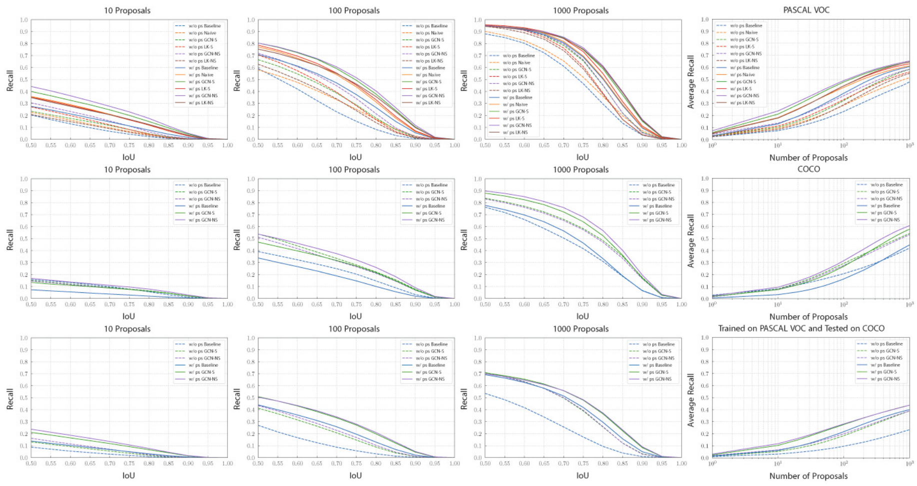

Fig. 3. Recall against IoU with different proposal numbers of 10, 100 and 1,000 and average recall against the number of proposals of all models: The results of PASCAL VOC 07test using models trained on PASCAL VOC 07+12 (**Row 1**). The results of COCO minival using models trained on COCO trainval35k (**Row 2**) and models trained on PASCAL VOC 07+12 (**Row 3**).

Table 2. Object proposal results of all models trained on PASCAL VOC `07+12` and evaluated on `07test`

	w/o position-sensitive							w/ position-sensitive						
	AR^{10}	AR^{100}	AR^{1k}	AUC	AR_s^{1k}	AR_m^{1k}	AR_l^{1k}	AR^{10}	AR^{100}	AR^{1k}	AUC	AR_s^{1k}	AR_m^{1k}	AR_l^{1k}
Baseline	.074	.234	.480	.272	.254	.414	.566	.131	.385	.605	.399	.423	.583	.655
Naïve	.094	.286	.515	.313	.410	.418	.596	.182	.434	.613	.435	.466	.593	.655
GCN-S	.103	.325	.584	.356	**.471**	.558	.623	.212	.479	.644	.471	.445	.603	.709
LK-S	.113	.333	.562	.356	.441	.547	.595	.178	.447	.630	.446	.463	**.613**	.674
GCN-NS	.136	.365	.586	.383	.445	.569	.625	**.238**	**.490**	**.653**	**.484**	.453	.593	**.730**
LK-NS	.084	.290	.551	.326	.420	.553	.575	.179	.447	.645	.450	.429	.582	.728

Table 3. Object proposal results of all models trained on COCO `trainval35k` and evaluated on `minival`

	w/o position-sensitive							w/ position-sensitive						
	AR^{10}	AR^{100}	AR^{1k}	AUC	AR_s^{1k}	AR_m^{1k}	AR_l^{1k}	AR^{10}	AR^{100}	AR^{1k}	AUC	AR_s^{1k}	AR_m^{1k}	AR_l^{1k}
Baseline	.083	.208	.421	.242	.308	.562	.392	.034	.165	.448	.219	.385	.411	.566
GCN-S	.082	.294	.542	.322	.414	.558	.680	.075	.270	.579	.321	.485	.592	.677
GCN-NS	.079	.277	.532	.310	.422	.552	.643	**.096**	**.316**	**.607**	**.358**	**.493**	**.623**	**.726**

AR_s^{1k} in particular benefits from learning the global image context. One can observe that compared to *Baseline* model, the scores have been boosted by 85% from 0.254 to 0.471 on PASCAL VOC with GCN-S model, and by 37% from 0.308 to 0.422 on COCO with GCN-NS. Therefore, the AR^{1k} has been overall improved from 0.480 and 0.421 to 0.586 and 0.542, which are 22% and 29% respectively. Fig. 3 also shows that GCN-S and GCN-NS models have the highest recall scores across all IoU thresholds with different proposal numbers.

One may argue that the improvement in GCN-S and GCN-NS models is due to the increased number of parameters. From Table 2, LK-S and LK-NS models have also shown some improvement but considering the extra number of parameters compared with *Baseline* model, they are not as effective as GCN-S and GCN-NS models. This shows that using separable convolution kernel matters, which aligns with the observation in [19]. *Naïve* model also exhibits similar results.

4.2 Impact of Using Position-Sensitivity

As shown in Table 2, on PASCAL VOC, among different proposal numbers and object sizes, models using position-sensitive components generally result in higher AR. Specifically, for AR^{1k}, *Baseline* model shows an improvement from 0.480 to 0.605 (26%) and GCN-NS from 0.586 to 0.653 (11%). As shown in Table 3, the experiment on COCO shows similar results, in which *Baseline* model has an improvement from 0.421 to 0.448 (6%) and GCN-NS model from 0.532 to 0.607 (14%). To investigate on the small object proposals, AR_s^{1k} reveal that training on a large number of annotated small objects in COCO indeed helps in higher AR_s^{1k} scores, compared with the results of PASCAL VOC counterpart.

GCN-NS has achieved much higher AR_s^{1k} (0.493) score, which is a 17% improvement compared to the counterpart. Fig. 4 visualises the distribution heatmap and hit-and-miss of top 1,000 proposals using GCN-NS models with and without taking position-sensitivity into account, in which the hits are with a threshold 0.7 for the ground truth IoU. One can qualitatively tell that by using the position-sensitivity, models can generate proposals closer to objects and thus result in more hits, especially for objects with extremely large or small sizes.

4.3 Inference Time

Introducing both GCN structure and position-sensitive score maps in the RPN head brings in more learnable parameters resulting in more computation. Beside the input image size which has a directly impact on the overall inference time, the number of anchors, the kernel size of GCN and the grid number k of position-sensitive score maps are also key factors. Table 1 lists the models' inference times averaged on all input images in both `07test` and `minival`, in which *Baseline* model shows a performance of 26.6/58.2 (denoted for `07test/minival`) ms. Adding the position-sensitive score maps (with grid size $k = 4$) takes extra 9.1/21.3 ms, 9.8/19.8 ms and 8.9/20.1 ms for *Baseline*, GCN-S and GCN-NS model respectively, which show a comparable time difference. In contrast, introducing the GCN structure in the *Baseline* model adds 7.7/18.1 ms for GCN-S model while additional 9.3/25.3 ms for GCN-NS model. This reveals that using non-shared smoother generally takes more times than using shared smoother does. GCN-NS with position-sensitivity, as the best performance model, has a running time 44.8 ms ($\sim$22 fps) for `07+12` and 103.6 ms ($\sim$10 fps) for `minival`.

4.4 Model Generalisation

To evaluate the generalisation ability of the proposed method, the models trained on PASCAL VOC `07+12` are used to evaluate on COCO `minival`. Note that compared to COCO (80 categories), PASCAL VOC is a much smaller dataset with limited object categories (20 categories) and almost all categories in PASCAL VOC are included in COCO. We separate the categories of COCO into two sets: *common* and *non-common*. Common categories are ones in PASCAL VOC, while non-common are unseen categories. In addition, image scenes in PASCAL VOC are relatively simple with less small objects annotations. It is therefore more challenging for a model learned from PASCAL VOC to performance object proposal on COCO images. The results are shown in Table 4 and Fig. 3 (**Row 3**). Surprisingly, the AR^{1k} of the best performance models, GCN-S and GCN-NS with position-sensitivity, trained with PASCAL VOC `07+12` can still outperform *Baseline* model trained with COCO `trainval35k` (i.e., 0.438 vs. 0.421). It is expected that the model will not work well on small objects since PASCAL VOC does not contain many small objects, but nevertheless the model still shows decent performance on large objects in which the AR_l^{1k} is up to 0.693 as shown in Table 4. The score is comparable to the other models trained on COCO `trainval35k` as shown in Table 3. Delving into the details, we

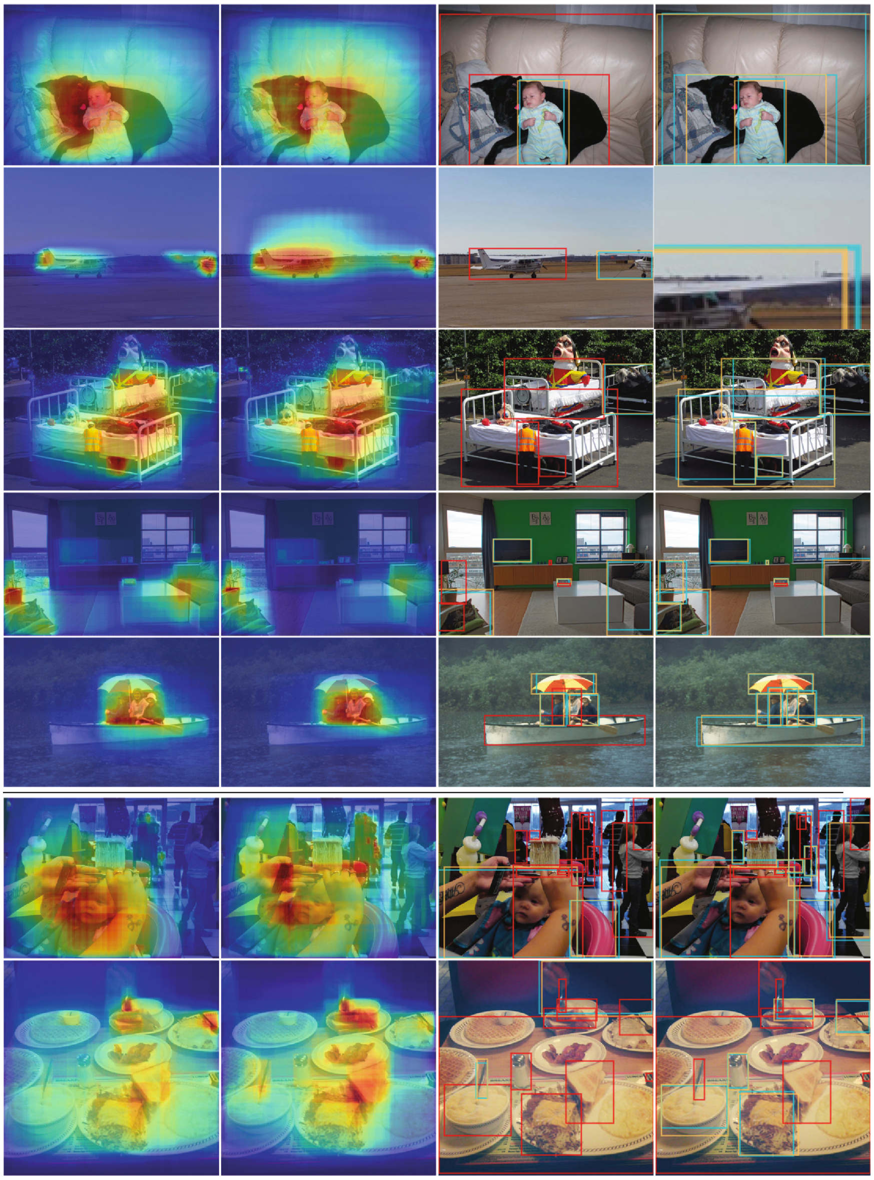

Fig. 4. The impact of position-sensitivity: visualisation on the distribution heatmap and hit-and-miss of the top 1,000 proposals by GCN-NS models for a number of selected PASCAL VOC `07test` (**Row 1–2**) and COCO `minival` (**Row 3–5**) images. For each pair of images, model without position-sensitivity is on the left and the one with position-sensitivity is on the right. **Col 1–2:** The heatmaps are plotted by stacking the proposal boxes. **Col 3–4:** The bounding boxes in orange are ground truth boxes with their corresponding hit proposals in cyan, in which the IoU threshold is set to 0.7, and the bounding boxes in red are missed cases. **Row 6–7:** All models tend to fail in images with complex scenes and diverse object aspect ratios. Nonetheless, note that models with position-sensitivity generally have higher hit rate. Sect. 4.2 for detailed discussions.

show the breakdown results of the model generalisation experiment for common and non-common categories are shown in Table 5. All models have better performance on common categories overall. However, compared to the *Baseline* model, the proposed components significantly improved the performance on both common and non-common categories. The per-category AUC performance in Fig. 5 shows that non-common categories do not necessarily have worse performance than common categories (e.g., bear, zebra, and toilet). This indicates that the proposed object proposal networks are able to generalise proposals from a smaller to a larger and more complex dataset, from common to non-common categories, and that the proposed architecture can further improve the generalisation for all categories. Fig. 6 qualitatively demonstrates the generalisation ability of the proposed method. Although the model trained on PASCAL VOC 07+12 fails at detecting small objectness, it still exhibits a certain degree of generalisation to unseen categories (e.g., elephant, teddy bear, or fire hydrant).

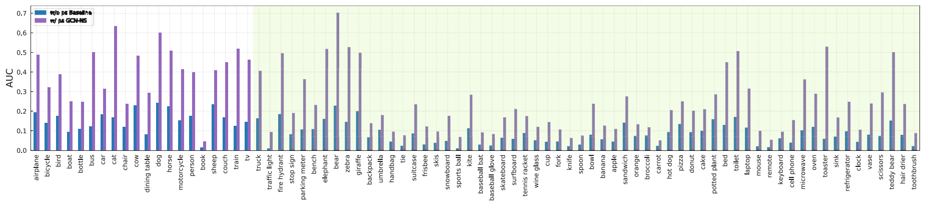

Fig. 5. Per-category AUC performance of models trained on PASCAL VOC 07+12 and evaluated on COCO `minival`. Common and non-common categories are split in the white and green region respectively.

Table 4. Object proposal results of all models trained on PASCAL VOC 07+12 and evaluated on COCO `minival`

	w/o position-sensitive							w/ position-sensitive						
	AR^{10}	AR^{100}	AR^{1k}	AUC	AR_s^{1k}	AR_m^{1k}	AR_l^{1k}	AR^{10}	AR^{100}	AR^{1k}	AUC	AR_s^{1k}	AR_m^{1k}	AR_l^{1k}
Baseline	.031	.097	.234	.122	.114	.234	.382	.061	.218	.400	.240	.224	.401	.614
GCN-S	.053	.185	.390	.217	**.240**	.430	.524	.104	.277	**.438**	.288	.227	**.463**	.665
GCN-NS	.066	.200	.390	.228	.239	.420	.538	**.118**	**.282**	**.438**	**.292**	.235	.432	**.693**

Table 5. Object proposal results of all models trained on PASCAL VOC 07+12 and evaluated on COCO `minival` for *common* and *non-common* categories. Note that (non) denotes models evaluated on *non-common* categories

	w/o position-sensitive							w/ position-sensitive						
	AR^{10}	AR^{100}	AR^{1k}	AUC	AR_s^{1k}	AR_m^{1k}	AR_l^{1k}	AR^{10}	AR^{100}	AR^{1k}	AUC	AR_s^{1k}	AR_m^{1k}	AR_l^{1k}
Baseline	.046	.129	.285	.156	.180	.316	.369	.076	.271	.483	.294	.329	.506	.629
Baseline (*non*)	.013	.055	.170	.079	.035	.139	.402	.043	.151	.295	.171	.099	.280	.590
GCN-S	.061	.217	.447	.252	.329	.499	.524	.128	.344	.520	.351	.327	**.564**	.687
GCN-S (*non*)	.042	.144	.317	.172	.134	.349	.525	.073	.192	.334	.207	.107	.346	.632
GCN-NS	.078	.240	.451	.269	.323	.493	.549	**.149**	**.353**	**.524**	**.360**	**.335**	.540	**.716**
GCN-NS (*non*)	.050	.150	.312	.175	.138	.334	.522	.078	.192	.328	.206	.114	.305	.658

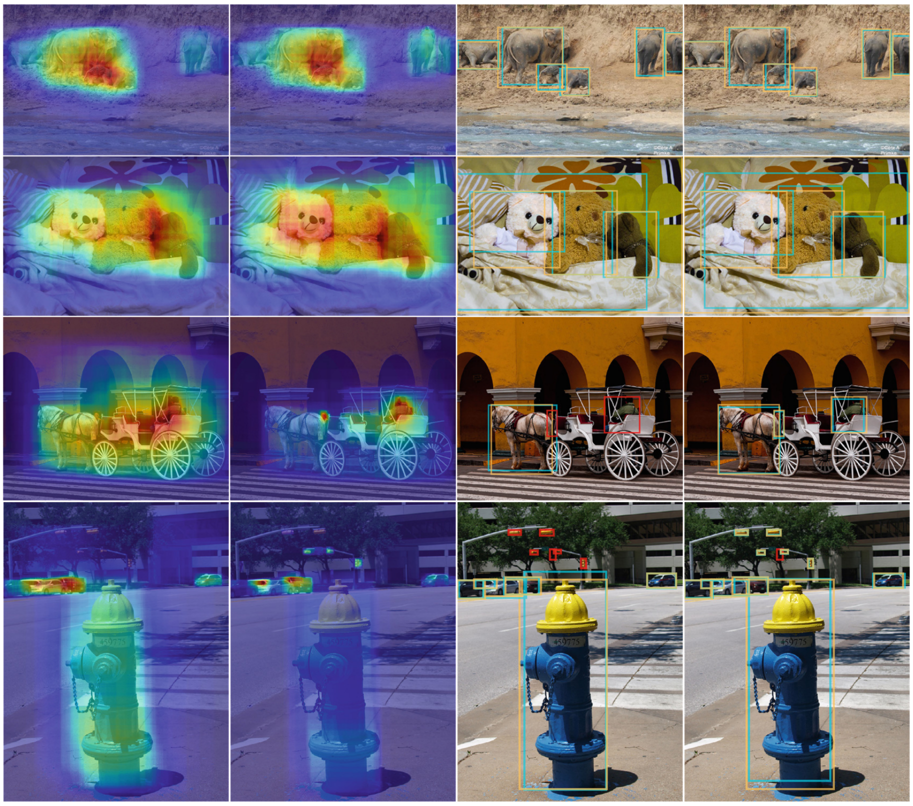

Fig. 6. The results of model generalisation experiments visualised in the distribution heatmap and hit-and-miss of the top 1,000 proposals by GCN-NS models for a number of selected COCO `minival`. For each pair of COCO `minival` images, result of model trained on PASCAL VOC is on the left and the one of model trained on COCO is on the right. See Sect. 4.4 for detailed discussions.

5 Conclusions

In this paper, we have proposed object proposal networks based on the observation that accurate detection relies on translation-invariance for objectness classification, translation-variance for localisation as regression and scale-invariance for various object sizes. Thorough experiments on PASCAL VOC and COCO datasets have shown that the adoption of global convolutional network (GCN) and position-sensitivity components can significantly improve object proposal performance while keeping the network lightweight to achieve real-time performance.

References

1. Arbeláez, P., Pont-Tuset, J., Barron, J.T., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: CVPR (2014)
2. Bell, S., Lawrence Zitnick, C., Bala, K., Girshick, R.: Inside-outside net: detecting objects in context with skip pooling and recurrent neural networks. In: CVPR (2016)
3. Carreira, J., Sminchisescu, C.: CPMC: automatic object segmentation using constrained parametric min-cuts. In: TPAMI (2012)
4. Chavali, N., Agrawal, H., Mahendru, A., Batra, D.: Object-proposal evaluation protocol is 'gameable'. In: CVPR (2016)
5. Dai, J., Li, Y., He, K., Sun, J.: R-FCN: object detection via region-based fully convolutional networks. In: NIPS (2016)
6. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The PASCAL visual object classes (VOC) challenge. IJCV **88**, 303–338 (2010)
7. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. TPAMI **32**, 1627–1645 (2010)
8. Gidaris, S., Komodakis, N.: Object detection via a multi-region and semantic segmentation-aware CNN model. In: ICCV (2015)
9. Girshick, R.: Fast R-CNN. In: ICCV (2015)
10. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR (2014)
11. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8691, pp. 346–361. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10578-9_23
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
13. Hosang, J., Benenson, R., Dollár, P., Schiele, B.: What makes for effective detection proposals? TPAMI **38**, 814–830 (2016)
14. Kong, T., Yao, A., Chen, Y., Sun, F.: Hypernet: towards accurate region proposal generation and joint object detection. In: CVPR (2016)
15. Kuo, W., Hariharan, B., Malik, J.: DeepBox: Learning objectness with convolutional networks. In: ICCV (2015)
16. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: CVPR (2017)
17. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
18. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
19. Peng, C., Zhang, X., Yu, G., Luo, G., Sun, J.: Large kernel matters-improve semantic segmentation by global convolutional network. In: CVPR (2017)
20. Pinheiro, P.O., Collobert, R., Dollár, P.: Learning to segment object candidates. In: NIPS (2015)
21. Pinheiro, P.O., Lin, T.-Y., Collobert, R., Dollár, P.: Learning to refine object segments. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9905, pp. 75–91. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46448-0_5

22. Pont-Tuset, J., Van Gool, L.: Boosting object proposals: from pascal to COCO. In: ICCV (2015)
23. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems (2015)
24. Ronneberger, O., Fischer, P., Brox, T.: U-Net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
25. Shrivastava, A., Gupta, A., Girshick, R.: Training region-based object detectors with online hard example mining. In: CVPR (2016)
26. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2014)
27. Uijlings, J.R., Van De Sande, K.E., Gevers, T., Smeulders, A.W.: Selective search for object recognition. IJCV **104**, 154–171 (2013)
28. Viola, P., Jones, M.J.: Robust real-time face detection. IJCV **57**, 137–154 (2004)
29. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8689, pp. 818–833. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10590-1_53
30. Zitnick, C.L., Dollár, P.: Edge boxes: locating object proposals from edges. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 391–405. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_26