A Lightweight Region Proposal Network for Task Specific Applications

Simon O'Keeffe
Department of Electronic Engineering
Maynooth University
Maynooth, Ireland
simon.okeeffe.2010@mumail.ie

Rudi Villing
Department of Electronic Engineering
Maynooth University
Maynooth, Ireland
rudi.villing@mu.ie

***Abstract*— Quickly and cheaply finding areas of interest within an image can save computationally intensive image processing in a vision pipeline. Existing region proposal networks are either too general (finding all objects in an image) or too complex (providing fine-tuned bounding boxes for classification). We propose a straightforward region proposal network that simply scores parts of the image based on whether or not they contain an object of interest. This calculation can be carried out quickly and for many applications including autonomous driving only a small fraction of the image area may contain objects of interest. We trained our network on an autonomous robot soccer dataset with similar characteristics to the popular KITTI autonomous driving dataset and achieved a recall greater than 95% while eliminating on average over 80% of the image area from further processing.**

Keywords—Region Proposal, convolutional Neural Networks, real-time, vision pipeline

I. Introduction

Real world image processing for tasks like autonomous driving, pedestrian detection, and service robots require solutions that are real-time capable. State-of-the-art deep learning techniques [1], [2] are computationally expensive and difficult to run in real-time without powerful GPUs. Quickly searching an image for a region of interest can eliminate the need to carry out extensive and expensive image processing on regions of the image where no objects of interest are present [3]. In many real-world problems, large areas of the image contain no objects of interest and hence no regions of interest for object detection. In autonomous driving, typical examples of this would be areas of empty road or sky. Quickly identifying the regions of interest (for object detection) within a given image can massively reduce the computation required to process an image.

In standard region proposal networks, regions are proposed with bounding box coordinates and a score indicating how likely it is that the bounding box contains an object [4]. This region proposal approach generates multiple overlapping bounding boxes as outputs, each of which are subject to further processing (e.g. for classification). Furthermore, these region proposal networks typically produce 10^3 proposals for a given image [3]. This leads to substantial processing being required further on in the image processing pipeline. In contrast to this we propose a simpler region proposal network that uses a fixed grid as the output and where each cell in the grid is activated only if an object of interest intersects with the cell.

Our straightforward region proposal network has the advantage that there will never be any overlapping regions proposed. Furthermore, the maximum number of proposals is N^2 where N is the side of the grid at the output layer. In this work we will analyse a popular deep learning autonomous driving KITTI dataset [5] to demonstrate that large portions of real-world image processing tasks contain little information. We will then show that this property is also reflected in our real-world dataset for autonomous robot soccer [6].

The remainder of this paper will cover existing work in the literature in Section II. We outline our proposed approach in Section III. Section IV details our results and Section V presents our conclusions.

II. Related Work

Before region proposal networks, object detectors relied on the well-known sliding window paradigm [7], [8] where a classifier would be run in every candidate image window. Sliding window classifiers are exhaustive, requiring around 10^4-10^5 windows per image at a single scale, with the number of windows increasing by an order of magnitude for multi-scale detection. The sliding window approach was acceptable provided that the classifier to run on each window was efficient [7] however modern object detectors use Convolutional Neural Networks [9] where the sliding window paradigm is much less computationally feasible.

RCNN replaced the sliding window paradigm with Selective Search [10]. Selective Search has no learned parameters and greedily merges superpixels to generate proposals. Selective search parameters are manually tuned for best performance. Similar and faster methods than Selective Search are Bing [11] and EdgeBoxes [12]. Bing uses a simple linear classifier trained over edge features and applied in a sliding window manner. EdgeBoxes also starts from a coarse sliding window pattern but builds on object boundary estimates and adds a subsequent refinement step to improve localisation. In both Bing and EdgeBoxes no parameters are learnt. The above proposal methods reduce the number of proposals from selective search from 10^6 proposals down to around 10^3 proposals. Furthermore, these methods are class agnostic detectors and work to propose any possible object. However, it has been shown that similar performance to the Bing method can be achieved without looking at the image. This image independent method, CrackingBing [13] achieved similar performance to Bing using combinatorial geometry and exploiting the evaluation framework used to score proposal networks.

When using class agnostic detectors, all objects are proposed for processing. Many real-world systems require proposals only for objects of interest to the application for efficient vision pipelines. For the problem of autonomous driving as few as two distinct classes for car and pedestrian are considered [5]. Deep Learning approaches can be used to train region proposal networks specific to a given problem. In Faster RCNN, the Selective Search algorithm was replaced by a Region Proposal Network [14] which was trained on the same dataset as the rest of the vision pipeline. The Region Proposal Network used in Faster RCNN used a state-of-the-art CNN [15] to generate proposals and shared the convolutional features extracted from the network for both the

S. O'Keeffe and R. Villing, "A Lightweight Region Proposal Network for Task Specific Applications," 2019 30th Irish Signals and Systems Conference (ISSC), Maynooth, Ireland, 2019, pp. 1-6, doi: 10.1109/ISSC.2019.8904949.
978-1-7281-2800-9/19/$31.00 ©2019 IEEE

region proposal and subsequent classification parts of the pipeline.

The Region Proposal Network in Faster RCNN is not class agnostic and is trained to propose regions for specific object classes. However it uses a computationally intensive CNN on the whole image. Unlike the Region Proposal Network in Faster RCNN, we propose a lightweight computationally inexpensive CNN for Region Proposal. By using a lightweight CNN in the early stage of the vision pipeline we can reduce the unnecessary computation on parts of the image where no objects are present. Our proposed network only outputs regions of interest in an image whereas Faster RCNN's Region Proposal Network proposes bounding boxes. Faster RCNN's Region Proposal Network operates on convolutional features generated by a state-of-the-art CNN [15] which runs at least 8×slower than the most computationally intensive YOLO network [2].

III. Proposed Approach

Our proposed approach depends on the hypothesis that in many real-world image processing problems only a small percentage of a given image actually contains an object or region of interest. To examine this, we first perform an analysis of a real-world autonomous driving dataset along with a dataset from a robot soccer problem. After this we give an overview of our proposed network for the region proposal problem.

A. Problem Analysis

In real world vision problems only a fraction of an image may contain regions of interest for a given problem. This includes problems such as autonomous driving where images contain areas of empty road, buildings or sky. The KITTI dataset [5] is a representative dataset used to evaluate object detection for the autonomous driving domain. Analysis of the ground truth data for the KITTI dataset indicated that the mean object area coverage of the image was 9%.

We performed the same calculation on our own SPL dataset for autonomous robot soccer [6], which consists of 4,140 images containing 12,048 object instances (3,973 ball, 3,944 robot, 2,534 goal post, and 1,597 penalty spot). The dataset was gathered from a range of Nao robot camera image logs in various locations and lighting conditions including our lab, RoboCup2016 indoor and outdoor, and RoboCup 2017. The SPL dataset's mean object area coverage was 7% which was similar to the KITTI measure.

Our network outputs a grid where each cell contains an objectness score for the cell. Any grid cell which intersects with the ground truth bounding box of an object should be detected as an object. The size of the grid cells used for the output affects the total area proposed as containing an object, with larger grid cells resulting in more of the image being detected as shown in Figure 1. Our proposed region proposal network takes an input image size of 288×288 pixels which is used in state-of-the-art object detectors [2]. By using pooling layers with a 2×2 kernel and a stride of 2, various output grid sizes can be used as detailed in Table I.

TABLE I. Output Grid Size for Various Pooling Layers

Input Size	Number of Pooling Layers	Spatial Reduction	Output Size
288×288	1	2×	144×144
288×288	2	4×	72×72
288×288	3	8×	36×36
288×288	4	16×	18×18
288×288	5	32×	9×9

For this work we analyse networks with an output grid of 18×18 and 9×9 as can be seen in Figure 1 for the KITTI dataset and Figure 2 for our SPL dataset.

The area of image covered by ground truth region increases as the grid cell size increases as can be seen in Table II.

TABLE II. Mean Area covered for Various Grid Sizes

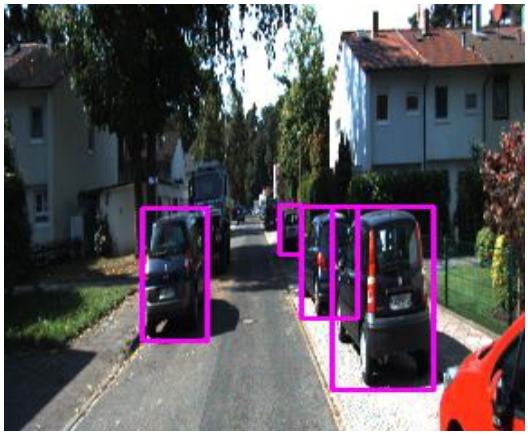
Output Size	Mean Area	Min Area	Max Area	Std Dev.
288×288	7%	0.06%	93%	9%
72×72	8%	0.2%	94.6%	9.2%
36×36	9%	0.3%	94.6%	9.5%
18×18	11.2%	0.3%	95.7%	10.1%
9×9	16.2%	1.2%	96.3%	11.6%

B. Prosposed network

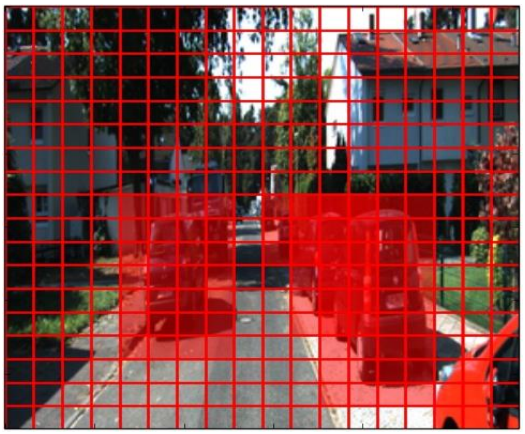
The proposed region proposal network uses standard convolution and pooling layers, the network is modelled on the tiny YOLO architecture [2]. We chose the tiny YOLO architecture as a starting point as it is one of the object detector networks with a low computational cost, requiring a relatively small number of FLOPs for a forward pass through the network. The number of FLOPs is given by the following equation.

$$FLOPS = 2HW(C_{in}K^2 + 1)C_{out} \quad (1)$$

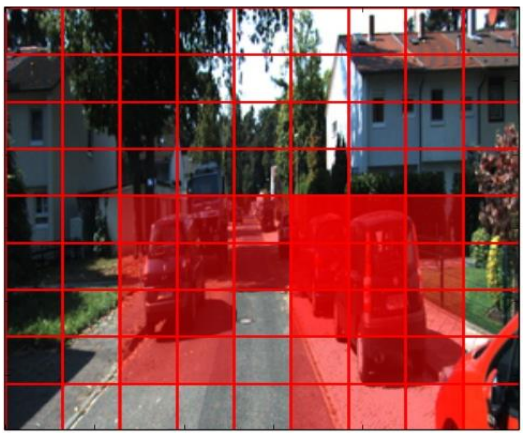
where H, W, and C_{in} are height, width and number of channels of the input feature map, K is the kernel size (assumed to be symmetric), and C_{out} is the number of output channels.



(a)

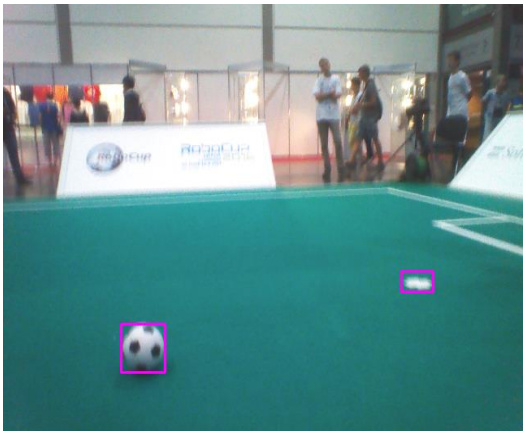


(b)

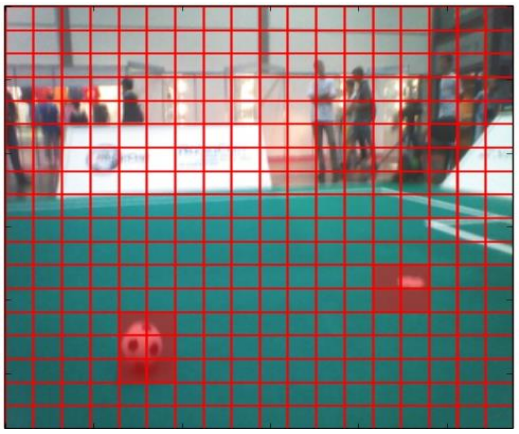


(c)

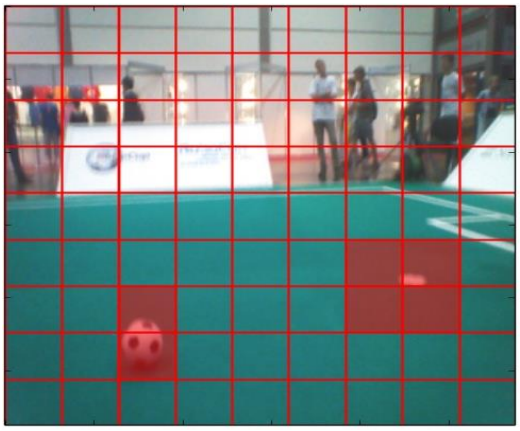
Fig. 1. Analysis of area coverage for proposed region proposal network for KITTI dataset with ground truth shown in (a) and area coverage for an 18×18 and 9×9 grid shown in (b) and (c) respectively.



(a)



(b)



(c)

Fig. 2. Analysis of area coverage for proposed region proposal network for SPL dataset with ground truth shown in (a) and area coverage for an 18×18 and 9×9 grid shown in (b) and (c) respectively.

For our networks we use the same architecture as tiny YOLO for the first 5 convolution layers with a reduced numbers kernels in each layer. This common architecture is detailed in Table III. The final layers of our two networks differ to output grid cells at a 9x9 and 18x18 resolution. In practice this simply means removing one extra pooling layer to have a larger spatial dimension at the output layer. The final layers for Region Proposal 18 and Region Proposal 9 networks are detailed in Table IV and Table V respectively. Comparing the computational load of both these networks to the tiny YOLO network trained for object detection on the same dataset, the Region Proposal 18 network results in a 6.1x reduction in computational cost while the Region Proposal 9 network results in a 11.3× reduction in computational cost. The tiny YOLO network took 2.56 G FLOPs for one forward pass through the network.

TABLE III. COMMON LAYERS OF OUR REGION PROPOSAL NETWORKS

Layer Type	Kernel Size / Stride	Kernels	Input Size	FLOPS
Conv	3×3 / 1	8	288×288×3	37.2 M
Pool	2×2 / 2		288×288×8	
Conv	3×3 / 1	16	144×144×8	48.4 M
Pool	2×2 / 2		144×144×16	
Conv	3×3 / 1	32	72×72×16	48.1 M
Pool	2×2 / 2		72×72×32	
Conv	3×3 / 1	64	36×36×32	47.9 M
Pool	2×2 / 2		36×36×64	
Conv	3×3 / 1	128	18×18×64	47.9 M
Total				229.5M

TABLE IV. REGION PROPOSAL 18 NETWORK ARCHITECURE

Layer Type	Kernel Size / Stride	Kernels	Input Size	FLOPS
Common Layers			18x18x64	229.5 M
Conv	3×3 / 1	256	18×18×128	191.1 M
Conv	1×1 / 1	1	18×18×256	0.167 M
Total				420.8 M

TABLE V. REGION PROPOSAL 9 NETWORK ARCHITECURE

Layer Type	Kernel Size / Stride	Kernels	Input Size	FLOPS
Common Layers			18x18x128	229.5 M
Pool	2×2 / 2		18×18×128	
Conv	3×3 / 1	256	9×9×128	47.8 M
Conv	1×1 / 1	1	9×9×256	0.042 M
Total				227.3 M

To train the region proposal network, we optimize the straightforward loss function:

$$Loss = \sum_{i=0}^{s^2} \mathbb{I}_i^{obj}(O_i - \hat{O}_i) + \lambda_{noobj} \sum_{i=0}^{s^2} \mathbb{I}_i^{noobj}(O_i - \hat{O}_i)$$

where s represents the spatial dimension of the output image, O_i represents the predicted objectness score and $\hat{O}_i$ represents the ground truth object score. $\mathbb{I}_i^{obj}$ and $\mathbb{I}_i^{noobj}$ is one whenever an object is present or not present for cell *i*. For a ground truth for a given cell *i*, $\hat{O}_i$ is 1 if the cell intersects with the ground truth bounding box of an object and 0 is the cell does not intersect with any object.

C. Post Network Processing

The output of our network is an objectness score for every output cell which can result in irregular contiguous regions being outputted as a single proposal. Most standard state-of-the-art image processing techniques [1], [2], [9], [14], [15] use a rectangular image as input and to facilitate this we expand out contiguous blobs to a rectangular shape for further processing. This expansion is done by creating a bounding box out of the edgemost sections of a contiguous region. Expanding a region in this way creates more of the image required for further processing but also increases the recall of the entire proposal network. Figure 3 shows the results of expanding out contiguous connected cells into rectangular sections.

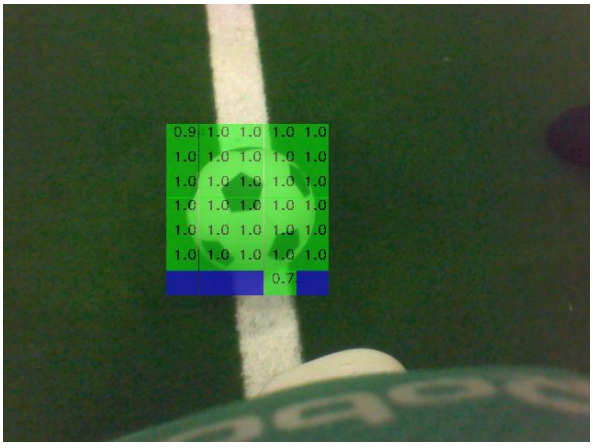


Fig. 3. Post network processing to expand connected regions into rectangles. The green area shows detected cells and the blue are shows detected cells added after expanding connected cells.

IV. RESULTS

We trained both our networks for 100,000 iterations with a batch size of 32 (determined by the computational capabilities of the evaluation platform). For training, our dataset is divided into training, validation, and test sets in an 80:10:10 split. The precision recall curves for both trained networks can be seen in Figure 4. The precision recall curve is calculated for confidence ranges from 5% to 95% at intervals of 0.5%. Both networks tended to score higher on precision than on recall. At a confidence threshold of 50%, Region Proposal 9 has a precision of 94.7% and a recall of

90.2%. Region Proposal 18 scores similarly with 94.4% precision and 89.9% recall. Choosing a lower confidence threshold increases network recall but decreases network precision with Region Proposal 9 reaching 92.8% precision.

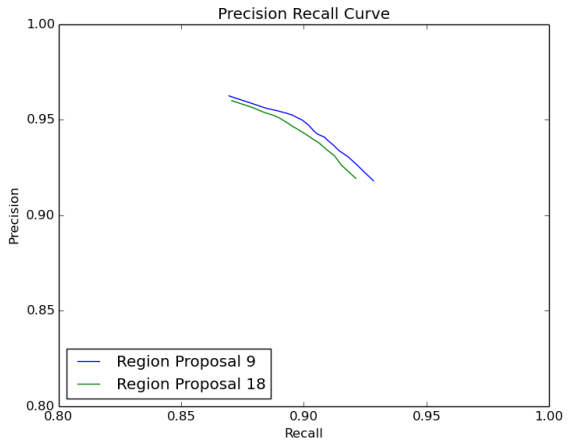


Fig. 4. Precision Recall curve for Region Proposal 9 and Region Proposal 18 networks.

Expanding detected cells into rectangular regions to allow for further processing increases the recall but decreases the precision. It also increases the area of the image which is then subject to further processing. The precision recall curve for the expanded regions are shown in Figure 5 which were calculated for confidence ranges from 10% to 90% at intervals of 10% for the Region Proposal 9 network. The recall for Region Proposal 9 reaches 95.2% at a confidence of 10% however precision drops to 80.2%.

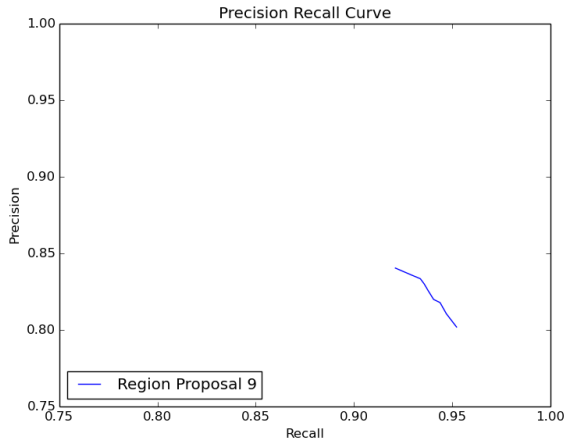


Fig. 5. Precision Recall curve for Region Proposal 9 network with detected cells expanded to rectangular regions.

The size of proposed regions of interest directly affects the computation required for further processing. An efficient region proposal network should score high on recall while proposing as small an area of the image as possible. The smallest region of interest area that it is possible to propose while having high recall is determined by the ground truth bounding box. The number of cells in the output grid also determines the minimum area that can be proposed while encompassing the ground truth bounding box. Our results indicated that a 9×9 output grid resulted in an average of 15.6% of an image being proposed as object cells while an 18×18 output grid proposed just 10.8% of the image area on average.

However, subsequent processing generally requires rectangular regions and therefore the smallest rectangle that can encompass the adjacent object cells must be generated. For the Region Proposal 9 network (with a 9×9 output grid) this increased the area of proposed regions from 15.6% to 18.6% on average. The computational load of the region proposal network can be traded against the total area of proposal regions by changing the output grid size. The Region Proposal 9 network has a lower computational load than the Region Proposal 18 network but yields proposals that cover more of the image area. When the region proposal network is incorporated in a full vision pipeline an optimisation of total computational load can be performed by trading computational load in the region proposal network (which is affected by output grid size) to computational load in the remainder of the pipeline (which depends on total area of regions to be processed further).

V. Conclusion

In this paper we have proposed a straightforward region proposal network for task specific image processing problems where only a small area of the image contains objects of interest which is a problem encountered in real world data sets including those for autonomous driving. Our proposed solution can has a recall of 95%,a precision of 80%, and proposes an average image area of 18.6%. For region proposal networks, a higher recall is more important than a higher precision as precision can be improved with further processing but recall cannot. Our network uses 11.3× fewer FLOPs than the tiny YOLO network while performing inference. Nevertheless, the tiny YOLO network performs full object detection and processes the entire image in one CNN. Although our approach only proposes regions of interest and must be integrated in an image pipeline that subjects proposed regions to further processing, it offers the possibility of faster object detection than tiny YOLO because it uses less than 9% of the computation of tiny YOLO. This leaves over 91% in computational bandwidth to process just 18% of the image. Therefore, less computation should be required to perform object detection overall and this is a direction of our future work.

Acknowledgment

The authors would like to gratefully acknowledge funding provided by the Irish Research Council under their Government of Ireland Postgraduate Scholarship 2013. This publication has emanated from research supported in part by a research grant from Science Foundation Ireland (SFI) under Grant Number 16/RI/3399.

References

[1] A. Krizhevsky, G. E. Hinton, and I. Sutskever, “ImageNet Classification with Deep Convolutional Neural Networks,” *Neural Inf. Process. Syst. Found. 2012 Conf.*, pp. 1097–1105, 2012.

[2] J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” *IEEE Conf. Comput. Vis. Pattern Recognit.*, pp. 6517–6525, Jul. 2017.

[3] J. Hosang, R. Benenson, and B. Schiele, “How good

are detection proposals, really?," Jun. 2014.

[4] J. Hosang, R. Benenson, P. Dollar, and B. Schiele, "What Makes for Effective Detection Proposals?," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 4, pp. 814–830, Apr. 2016.

[5] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[6] S. O'Keeffe and R. Villing, "A Benchmark Data Set and Evaluation of Deep Learning Architectures for Ball Detection in the RoboCup SPL," in *RoboCup International Symposium*, 2017.

[7] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, p. I-511-I-518.

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part-Based Models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[10] J. Uijlings, K. Van De Sande, and T. Gevers, "Selective search for object recognition," *Int. J.*, 2013.

[11] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr, "BING: Binarized Normed Gradients for Objectness Estimation at 300fps." pp. 3286–3293, 2014.

[12] C. L. Zitnick and P. Dollár, "Edge Boxes: Locating Object Proposals from Edges," Springer, Cham, 2014, pp. 391–405.

[13] Q. Zhao, Z. Liu, and B. Yin, "Cracking BING and Beyond," *BMVC*, pp. 1–10, 2014.

[14] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *Nips*, pp. 1–10, 2015.

[15] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *Int. Conf. Learn. Represent.*, pp. 1–14, 2015.