Weakly Supervised Semantic Roadside Object Segmentation Using Digital Maps

Johannes A.P. Guelen*†, Albert Ali Salah*, Bastiaan J. Boom† and Julien A. Vijverberg†
*Information and Computing Sciences, Utrecht University, Utrecht, the Netherlands
Email: jeroengoelen@gmail.com; a.a.salah@uu.nl
†CycloMedia Technology B.V., Utrecht, the Netherlands

***Abstract*—Publicly available digital maps may offer semantic information regarding objects in street view images. In this paper, we propose an approach to exploit such information to automatically create object detection datasets on which state-of-the-art object detection methods can be trained. To accomplish this, we use two detailed maps of the Netherlands containing the location of a large number of street objects. We link the object information to street view images to use them as image-wide labels. Our results show that even though there are many sources for noise in the labels, we can create useful data with this approach.**

***Index Terms*—Learning, Maps, Deep learning, Class activation maps, CAM loss, semantic segmentation, Roadside object detection**

1. Introduction

The ability to automatically detect and classify objects in images has led to increased capabilities of automatic systems, allowing them to solve increasingly complex real-world tasks. One of these complex tasks is the ability to detect and classify objects in a road environment, a critical task for autonomous driving and surveying applications. However, the number of possible roadside objects is high, and the appearance of these objects depends on the geographical location (e.g. see Fig. 1). A deep neural network to classify all possible objects requires a large training set and accompanying annotations.

In this paper, we investigate the possibility of using the semantic information from digital maps to automatically obtain image-wide labeling of roadside objects, thereby decreasing the cost of creating such training sets with manual object label annotations. Our approach is inspired by the work of Zadrija et al., where a sequenced image-wide labeled training set was constructed using Streetview footage and maps [1]. In this work, we investigate whether labeling roadside image with maps can enable roadside object detection for a large number of static objects. Furthermore, we investigate a method of acquiring semantic segmentation results from such an image-wide labeled dataset, based on Class Activation Maps (CAMs).

Figure 1: Examples of intra-class variance within the streetlight class

In an image-based classification task, a CAM is a map that shows which features and locations in the image will lead to a positive classification for a given class label. These maps are generated by looking back through the network's activations, which is possible if no fully connected layers are present in the classifier architecture. Subsequently, CAMs are useful tools in the analysis of convolutional neural network architectures. However,

978-1-7281-9331-1/20/$31.00 © 2020 IEEE

CAMs focus explicitly on the discriminative parts of objects [2]. To incorporate this representation in the loss function, we propose in this paper a CAM-loss that evaluates the CAM by comparing it to an approximate ground truth mask. While not precise, these approximation guides and enlarges the CAM, thereby including less discriminative parts of the objects in the computation of the loss. The mask itself is generated by using depth and location information available from digital maps. As a summary, our contribution in this paper is a method to generate an annotated roadside object database from digital maps and street view images, as well as methodological extensions to perform learning with such a resource.

2. Related Work

The performance of a state-of-the-art object detector is heavily influenced by the quality and the size of the training dataset. The cost of creating large annotated datasets is high, especially when labels are produced at semantic and/or instance level. Weakly supervised learning approaches use image-wide labels (which indicate the presence of an object, but not its location) to guide object detection tasks, and since there are large repositories and image search engines that allow images to be retrieved with such labels, this is a potentially useful solution to cut manual labeling costs down [3], [4]. It is also possible to combine image datasets with other information, such as maps that contain labels. Zadrija et al. proposed to merge Streetview footage and map information [1]. We adapt this idea in this paper and investigate whether a broader approach is possible for roadside object detection.

There are two weakly supervised learning approaches in the literature for object localization, namely, Multiple Instance Learning (MIL) [5], [6], [7], [8], [1], [9], [10] and localization via convolutional neural networks (CNN) [11], [12], [13], [14], [2], [15]. In MIL, data are grouped into bags, and if a bag contains at least one instance of the object, it receives a positive label for that object. For weakly supervised object detection with MIL, each bag consists of multiple regions of a single image [7], [6]. MIL is typically used for locating object boundaries with bounding boxes, and will not easily lead to full instance-level segmentation.

Localization via CNN can provide a more detailed segmentation of objects [11]. After a CNN is trained, the key idea is to look back through the CNN layers to see which areas led to a positive classification, and use this information for segmentation. Since fully connected (FC) layers remove the ability to localize objects, Zhou et al. proposed using a Global Average Pooling (GAP) layer to replace the FC layers [14]. This allowed them to create class activation maps (CAM) that show the location of the features that led to a positive classification. Recent approaches focus on ways to refine such CAMs for better segmentation [16].

In this paper, we investigate whether CAMs can be used during training and directly contribute to the loss function for object detection. More specifically, our intuition is that if a CAM is fixated on the wrong object, it should not contribute a small loss value, even if the label for the whole image is correct. However, our main assumption is that we do not possess ground truth labels at this level of detail.

3. Resources and Materials

In this section, we describe the resources we use for automatic dataset generation and for the creation of approximate ground truth. While our set of resources comes from the Netherlands, similar resources exist for other countries, and our approach can be used for creating similar datasets.

We use the Basic Large-scale Topographical Information (basis grootschalige topografie - BGT) maps and the Digital Topographical File (digitaal topographish bestand - DTB) maps in our research. These publicly accessible maps show the locations of millions of objects across the Netherlands. The BGT map shows all municipality owned objects, and the DTB map shows all objects owned by Rijkswaterstaat, the department overseeing highways, waterways and roads not controlled by municipalities. By combining these two maps, we gain access to around 184 different object types with an average of 62k objects per class. However, the database is severely unbalanced, and there are overlaps between the two maps. Table 1 lists the most frequently occurring objects in these maps. We select a subset of the more important objects for our experiments.

BGT-object	Amount	DTB-object	Amount
Drain	2550088	Map mark.	1124343
Streetlight	1983625	Illum. obj.	255984
Manhole	1266851	Drain	236510
Barrier post	509430	Traffic sign	144624
Traffic sign	229045	Bollard pole	106911
Traffic sign pole	197055	Hectometer sign	82356
Fire hydrant	170810	Manhole-cover	79874
Playground eq.	105821	Bollard rock	55370
Bicycle rack	105265	Utility cabinet	54059
Garbage can	94533	Roadside block	31168
Water well	87745	Paint sign	27876
Bench	83727	Signpost	25469
Waste can	52847	Traffic light	20160

TABLE 1: Top 13 most occurring objects from the BGT and DTB maps.

Figure 2: 2d representation of a cyclorama.

To match street view images for the locations indicated by the BGT and DTB maps, we use the Cyclomedia repository of images. Cyclomedia is a street view image company that has cars driving around the Netherlands (and other countries) with cameras taking images every 5 meters. Each location has six individual images, with 512×512 pixel resolution. These six images are combined to create a cyclorama (see Fig. 2 for a 2D representation of a cyclorama). These cycloramas are produced for each specific geographical location, and allow us to compute images for arbitrary view directions.

For some regions, the depth information is also available. This depth information is created by a Lidar system, which produces a gray-scale image, where the brightness of a pixel determines the distance of that pixel to the camera. These depth images directly overlap with the street-view images, as they are taken at the same time and position. Since many street view image acquisition systems also acquire such depth images, we explore whether additional improvements can be obtained by incorporating them in our analysis.

4. Methodology

In this section, we first describe our method of automatically generating a dataset from maps and images (Section 4.1). Then we detail our proposed method for image-wide object detection and semantic segmentation (Section 4.2). Finally, we go over our approach of improving the results by using automatically generated masks (Section 4.3).

4.1. Automatic Dataset Generation

An overview of the automatic training set generation method can be seen in Fig. 3. First, we take the BGT and DTB maps (A) and transform these into a list of coordinates of objects (B). A toroidal region (C) is described on the map around the coordinates of each object with a range of three to ten meters. For every object, all images in these zones are downloaded until a certain number is reached. The selected images are retrieved from the repository (D), and the object label is applied (E). To improve the precision of object labeling, a view cone is positioned with a field of view (FOV) of 90°(F). To find the other objects present in the images, we use the view cone with two pre-selected distances (X_{min} and X_{max}, shown in F as x and X). All objects up to X_{max} are considered visible, and all objects between X_{min} and X_{max} are additionally considered as hard to detect. These labels are added to the images to produce the automatically annotated dataset (G).

We focused on objects that can be represented by points (rather than lines and regions), and on roadside objects in particular. We fused classes in BGT and DTB maps when necessary (e.g. the class 'street light in the BGT maps is called 'illumination object in the DTB map). There are small differences within related class labels. For instance, 'illumination object also includes light elements that do not have a pole associated with them,

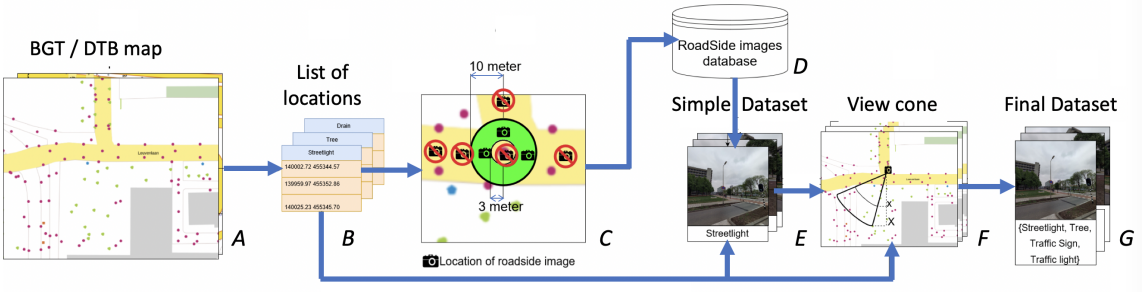

Figure 3: Overview of the automatic dataset generation method. See text for details.

(a) CAI cabinet (b) Cabinet house (c) Cabinet house

Figure 4: Images of two visually similar types of utility cabinets.

while the 'street light class only contains lights that are on top of street poles. Furthermore, some objects are too similar for a clear visual distinction. Figure 4 shows the difference between two types of 'utility cabinets, which can hardly be separated visually.

We investigate the performance of the method by selecting 20 objects of different shape, size, and rarity. With these objects, a training set of 16,000 images (800 images per class) and a validation set of 4,000 images (200 images per class) is created. It should be noted that some of the chosen object classes actually consist of multiple classes in the maps. These were combined as they were too similar to visually distinguish (e.g. in the case of utility cabinet, they consist of the classes 'Cabinet house', 'Communication Cabinet' and 'Electronic Cabinet'). We manually annotated a set of 400 images from the dataset to serve as a ground truth. The distances of the objects within the view cone were grouped into classes of very small ($X_{min} = 5$,$X_{max} = 10$), small ($X_{min} = 10$,$X_{max} = 15$), medium ($X_{min} = 20$,$X_{max} = 30$), and large ($X_{min} = 35$,$X_{max} = 45$), respectively. The object classes were allocated to one of these groups.

4.2. Object Detection and Segmentation

We use a ResNet-50 convolutional deep neural network trained on ImageNet as our baseline object detection framework [17]. We also experimented with a VGGnet-16 architecture [18], as this was originally used for the CAM method. However, newer methods [11] show increased performance on the ResNet-50 network and we found similar results for overall classification in our preliminary experiments. For object detection, the output layer of the ResNet-50 architecture is changed from a softmax layer to a sigmoid dense layer for the multi-label detection problem, with the size equal to the number of classes. During fine-tuning, a batch size of four is used, and different learning rates were tested. We used the decay and momentum values suggested by He et al. [17]. We kept training until our validation loss stagnated for more than four epochs. The epoch with the lowest validation loss is selected.

We propose to use a binary cross-entropy loss for training, as it is often used in multi-label classification. For objects in the hard zone, as described in Fig. 3, the loss is set to zero for that sample. The mean loss is computed per batch, and class-weights are added to reduce the effect of class imbalance by reducing the networks incentive to focus on the over-represented classes.

4.2.1. Localization of Objects with CAMs. To be able to localize an object precisely, we first detect it using the network described. We choose the CAM generation method, as introduced by Zhou

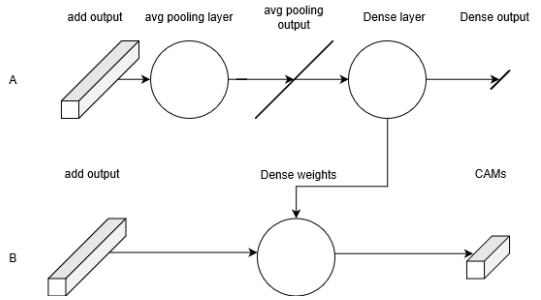

Figure 5: Image A shows a simplified diagram of the last layers of a ResNet-50 network, and Image B shows how to use those layers to generate CAMs.

et al. [14], over MIL-based approaches for precise localization. The latter need very accurate labels and are sensitive to label noise [5]. Furthermore, sliding window approaches are used to generate bags, and these do not work well with objects of very different sizes and shapes.

Generating a CAM from a network like ResNet-50 is simple, as the network already maintains localization information within its layers. We investigate CAM generation using the last convolutional layer and the last 'add' layer of the network. For a given input image, the output of the last convolutional layer (or the add layer) has a shape of $(16, 16, 2048)$. For every class, we find the values that lead to a positive classification. To accomplish this, the weights from the dense layer (with dimensionality $(n\_class, 2048)$) are used. By taking the dot product between these weights and the convolutional layer, the labels are directly connected to the convolutional map, resulting in a CAM for each class, shaped $(n\_class, 16, 16)$. Figure 5 illustrates this process and shows why the resolution of the CAM is directly linked to the size of the convolutional layer.

4.2.2. Semantic Segmentation. We investigate two methods to go from the CAMs to segmentation masks: The first is to simply use a high threshold on the normalized CAM values and use that as our segmentation mask. The second option is to use GrabCut [19], where the highest valued parts of the CAM are used to initiate the foreground area. Both methods require appropriate threshold values to be selected.

4.3. Automatically Generated Masks

In this section, we discuss whether the method of using available depth and map information can help guide the learning process and increase the quality of CAMs. First we discuss how this form of information can be used to create approximate object masks. Then we discuss how these can be used to train the network. Finally, we investigate how these masks can increase the resolution of the CAMs.

4.3.1. Approximate Mask Generation. We investigate improving CAM generation by introducing a CAM loss that penalizes focusing on incorrect image parts. As no direct object-masks are available within the dataset, an approximate ground truth mask is constructed with the help of the depth images available from the CycloMedia Lidar dataset. We use the relative direction of the object within the image (available from the object location and image location), as well as the distance to the object (see Fig. 6).

To create these masks, first an approximate depth range dr is calculated. Using the distance di between the object and the recorder's world location (image location), a range rw is selected. If the object is further away, the size of the range is increased. We use the following relation:

$$rw_i = max(di_i * o, 0.5), \tag{1}$$

where o is a variable greater than 0. To further specify the masks, a direction range dr for each object is also taken into account. First the relative direction of the object θ_d is calculated with regards to the recorder's world location b:

$$\theta_d = atan(ob_x - b_x, ob_y - b_y) - b_d, \tag{2}$$

where b_d is the direction d of the image with regard to north, and where ob_x, ob_y, b_x and b_y are the coordinates of the object and the image, respectively. A range ra is then set on this direction, taking into account the relation between the distance to the object and the size of the object:

$$ra_i = di_i * p \tag{3}$$

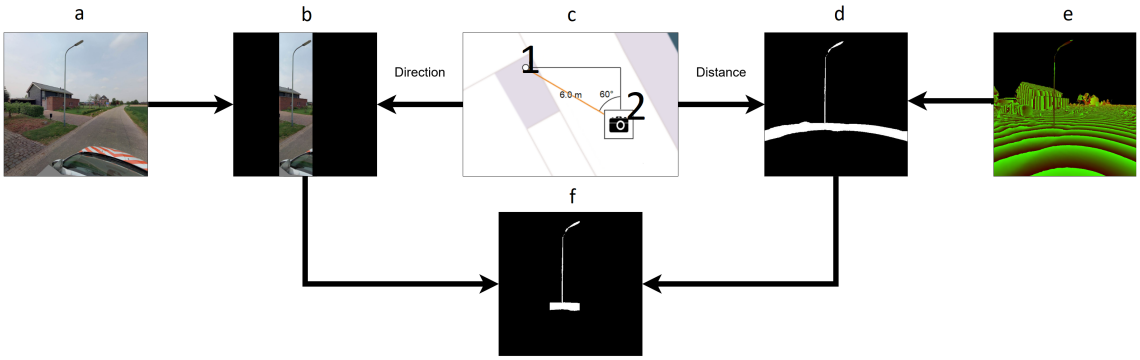

Figure 6: Overview of our proposed CAM loss method. To generate an approximate mask, the directions of all objects of a certain class are calculated first. Then the distance to all instances of this object is calculated, and a range is defined This is then combined with a range over the direction, resulting in approximate ground truth masks. The CAM loss is obtained by comparing this approximate ground truth mask with the CAM result using cross-entropy.

where p is a variable greater than 0. This range is then applied against the direction of the object:

$$dr_i = (\theta d_i - ra_i, \theta d_i + ra_i) \quad (4)$$

. These two ranges are then combined to generate a mask for every labeled instance visible in the image. We determined o and p empirically.

The masks are generated for every labeled object within an image, and then combined into a single mask per class, which is used during training and compared against a normalised CAM. Since these masks have a size of (512×512) pixels, CAMs are up-scaled to match them using bi-linear up-scaling (see Fig. 7).

4.3.2. Including CAM Loss within Training. To see the impact of the interaction between the two different loss functions (i.e. the normal cross-entropy loss function and the CAM loss function), multiple combined loss functions were investigated. Define l_1 to be the normal cross-entropy loss:

$$l_1 = -ylog(\hat{y}) - (1-y)log(1-\hat{y}), \quad (5)$$

where y and $\hat{y}$ are the ground truth label and the predicted label, respectively.

l_2 is the CAM loss, and consists of a cross-entropy comparison between the CAM and the approximate depth mask:

$$l_2 = -clog(\hat{c}) - (1-c)log(1-\hat{c}), \quad (6)$$

where c and $\hat{c}$ are the approximate depth mask and the CAM prediction, respectively. Combining the two losses can be accomplished by a weighted sum: $loss = w_1 l_1 + w_2 l_2$. We also investigated the relative loss function $loss = l_1 + l_1 l_2$ to reduce the effects of noise in the dataset. This relative loss function weights the CAM loss with the predicted label l_1.

As our model now has a second form of ground truth that directly gives us a loss on the CAMs, some up-scaling techniques can be used to increase the resolution of the CAM. This is done by adding a binary up-scaling layer before the last ResNet residual block. While the label loss cannot discriminate between these up-scaled values, the depth mask loss can. The downside is that the size of the model and the memory cost will increase. We selected a compromise and investigated up-scaling to 32×32 and to 64×64. Since CAM creation is noisy, up-scaling decreased the performance.

5. Results and Conclusions

To evaluate the quality of the generated set, we compared it to our manually annotated labels for the same images. We computed results for a range of X_{min} values, as this parameter influences the number of objects taken into account during training. Figure 8 illustrates the results in terms of F-score for all classes, as well as illustrating how

Figure 7: CAM for a street-gully.

F-score changes with X_{min} for four representative classes. Since these are image-level results, having multiple objects from the same class does not have an effect on the label. At a 10m range, the most difficult object to label is the information sign (F-score of 0.26), and the easiest objects are roadside protection block, bicycle stand, barrier post, and roadsign direction arrow (F-score over 0.75).

The results show that it is beneficial to select a range for every object class, and not use a generic range. Most of the relevant (visible) images are found within the 3-10 meters range, and this is also reflected in the results. The automatically generated labels produced an F-score of 0.66 on the average for the 20-class dataset, measured on the manually annotated portion.

The space and time complexity added to the baseline by the proposed approach is reasonable. The original CAM model required 2.9GB of space for parameters, which is increased to 3.1GB for the depth-enhanced model. The training time per batch is doubled (16s vs. 30s), but this is an offline cost. For predictions, one image takes 0.18s with CAM and 0.20s with the depth-enhanced model. Times are reported on non-optimized code on a machine with two Intel Xeon CPUs E5-2630 and two NVIDIA Tesla P100 12 gb GPUs.

We set out to investigate if digital maps could be used to construct an image-wide labeled dataset. With our proposed approach, regardless of the high amount of noise arising from the automatic generation process, a ResNet-50 network was capable of producing an average F-score of 0.69 when fine-tuned on the automatically generated training set. We observed that larger objects, such as barriers and bicycle stands, were better classified, whereas small objects, such as street gullies, were difficult to locate automatically.

Generating semantic masks for objects with this approach seems to be very difficult. The CAM loss we have introduced was able to slightly improve the ResNet-50 baseline, but further methodological progress seems to be required, and overall performance is very low for segmentation.

For image-level annotations, the most important issue is the large amount of label noise, and semi-supervised approaches may help in dealing with this problem. In our approach, we did not consider objects being blocked by other objects that are also labeled on maps. Known object relations could be used to improve visibility considerations. For example, if a street gully is behind a row of houses, it will most probably not be visible to the camera. Finally, the contextual information (presence of objects given other objects) can be incorporated into the models.

References

[1] V. Zadrija, J. Krapac, S. Šegvić, and J. Verbeek, "Sparse weakly supervised models for object localization in road environment," *Computer Vision and Image Understanding*, vol. 176, pp. 9–21, 2018.

[2] J. Ahn and S. Kwak, "Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation," in *Proc. CVPR*, 2018, pp. 4981–4990.

[3] E. J. Crowley and A. Zisserman, "The art of detection," in *Proc. ECCV*. Springer, 2016, pp. 721–737.

[4] K. Lai and D. Fox, "Object recognition in 3d point clouds using web data and domain adaptation," *The International Journal of Robotics Research*, vol. 29, no. 8, pp. 1019–1037, 2010.

[5] H. O. Song, R. Girshick, S. Jegelka, J. Mairal, Z. Harchaoui, and T. Darrell, "On learning to localize objects with minimal supervision," *arXiv preprint arXiv:1403.1024*, 2014.

[6] H. Bilen and A. Vedaldi, "Weakly supervised deep detection networks," in *Proc. CVPR*, 2016, pp. 2846–2854.

[7] R. G. Cinbis, J. Verbeek, and C. Schmid, "Weakly supervised object localization with multi-fold multiple instance learning," *IEEE TPAMI*, vol. 39, no. 1, pp. 189–203, 2016.

[8] X. Liang, S. Liu, Y. Wei, L. Liu, L. Lin, and S. Yan, "Towards computational baby learning: A weakly-supervised approach for object detection," in *Proc. ICCV*, 2015, pp. 999–1007.

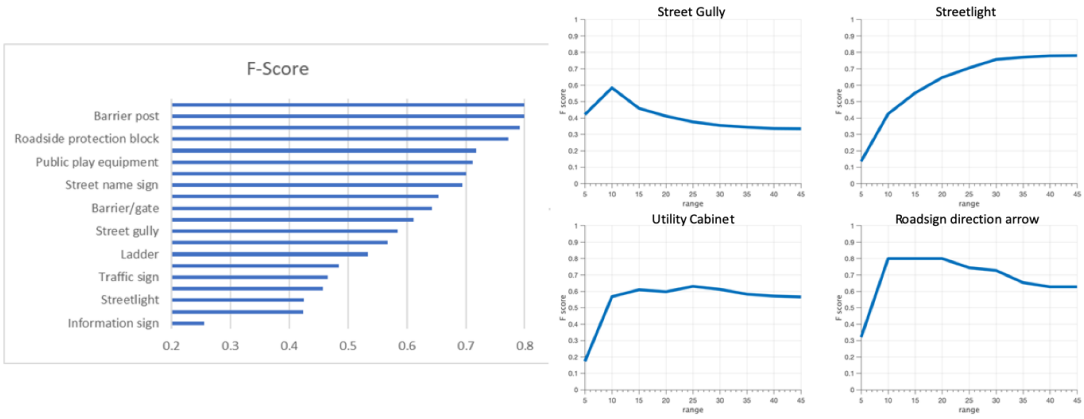

Figure 8: Left: F-scores for all classes. Right: F-scores vs X_{min} range for four representative classes.

[9] J. Wu, Y. Yu, C. Huang, and K. Yu, "Deep multiple instance learning for image classification and auto-annotation," in *Proc. CVPR*, 2015, pp. 3460–3469.

[10] P. Tang, X. Wang, X. Bai, and W. Liu, "Multiple instance detection network with online instance classifier refinement," in *Proc. CVPR*, 2017, pp. 2843–2851.

[11] Y. Zhou, Y. Zhu, Q. Ye, Q. Qiu, and J. Jiao, "Weakly supervised instance segmentation using class peak response," in *Proc. CVPR*, 2018, pp. 3791–3800.

[12] Y. Wei, Z. Shen, B. Cheng, H. Shi, J. Xiong, J. Feng, and T. Huang, "Ts2c: Tight box mining with surrounding segmentation context for weakly supervised object detection," in *Proc. ECCV*, 2018, pp. 434–450.

[13] Z. Jie, Y. Wei, X. Jin, J. Feng, and W. Liu, "Deep self-taught learning for weakly supervised object localization," in *Proc. CVPR*, 2017, pp. 1377–1385.

[14] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. CVPR*, 2016, pp. 2921–2929.

[15] J. Fan, Z. Zhang, and T. Tan, "CIAN: Cross-image affinity net for weakly supervised semantic segmentation," *arXiv preprint arXiv:1811.10842*, 2018.

[16] P. Du, H. Zhang, and H. Ma, "Classifier refinement for weakly supervised object detection with class-specific activation map," in *Proc. ICIP*. IEEE, 2019, pp. 3367–3371.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE TPAMI*, vol. 37, no. 9, pp. 1904–1916, 2015.

[18] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[19] C. Rother, V. Kolmogorov, and A. Blake, "Grabcut: Interactive foreground extraction using iterated graph cuts," in *ACM Trans. on graphics*, vol. 23, no. 3. ACM, 2004, pp. 309–314.