A Multistage Framework for Detection of Very Small Objects

Duleep Rathgamage Don
Kennesaw State University, drathgam@students.kennesaw.edu

Ramazan Aygun
Kennesaw State University, raygun@kennesaw.edu

Mahmut Karakaya
Kennesaw State University, mkarakay@kennesaw.edu

A Multistage Framework for Detection of Very Small Objects

D.R. Don*

School of Data Science and Analytics, College of Computing and Software Engineering, Kennesaw State University, Marietta, GA 30060, USA, drathgam@students.kennesaw.edu

R. Aygun

Department of Computer Science, College of Computing and Software Engineering, Kennesaw State University, Marietta, GA 30060, USA, raygun@kennesaw.edu

M. Karakaya

Department of Computer Science, College of Computing and Software Engineering, Kennesaw State University, Marietta, GA 30060, USA, mkarakay@kennesaw.edu

Small object detection is one of the most challenging problems in computer vision. Algorithms based on state-of-the-art object detection methods such as R-CNN, SSD, FPN, and YOLO fail to detect objects of very small sizes. In this study, we propose a novel method to detect very small objects, smaller than 8×8 pixels, that appear in a complex background. The proposed method is a multistage framework consisting of an unsupervised algorithm and three separately trained supervised algorithms. The unsupervised algorithm extracts ROIs from a high-resolution image. Then the ROIs are upsampled using SRGAN, and the enhanced ROIs are detected by our two-stage cascade classifier based on two ResNet50 models. The maximum size of the images used for training the proposed framework is 32×32 pixels. The experiments are conducted using rescaled German Traffic Sign Recognition Benchmark dataset (GTSRB) and downsampled German Traffic Sign Detection Benchmark dataset (GTSDB). Unlike MS COCO and DOTA datasets, the resulting GTSDB turns out to be very challenging for any small object detection algorithm due to not only the size of objects of interest but also the complex textures of the background. Our experimental results show that the proposed method detects small traffic signs with an average precision of 0.332 at the intersection over union of 0.3.

CCS CONCEPTS • ARTIFICIAL INTELLIGENCE • COMPUTER VISION • COMPUTER VISION PROBLEMS • OBJECT DETECTION

Additional Keywords and Phrases: Object detection, Computer vision, Deep learning, Image processing

ACM Reference Format:

First Author's Name, Initials, and Last Name, Second Author's Name, Initials, and Last Name, and Third Author's Name, Initials, and Last Name. 2018. The Title of the Paper: ACM Conference Proceedings Manuscript Submission Template: This is the subtitle of the paper, this document both explains and embodies the submission format for authors using Word. In Woodstock '18: ACM Symposium on Neural Gaze Detection, June 03–05, 2018, Woodstock, NY. ACM, New York, NY, USA, 10 pages. NOTE: This block will be automatically generated when manuscripts are processed after acceptance.

* Place the footnote text for the author (if applicable) here.

1 INTRODUCTION

Object detection comprises two major tasks known as identifying objects in an image and predicting their bounding box coordinates. Thus, object detection can be considered as a combination of classification and regression problems. Although object detection has been successful in diverse domains such as face recognition or autonomous driving, traditional approaches often fail to detect very small objects.

The detection of small objects with correct class and precise location is mainly affected by intraclass variation, variations in object rotation and scale, occlusion, and change in illumination [1]. Most of these challenges for regular-sized objects are reasonably handled by modern state-of-the-art object detection algorithms such as R-CNN, SSD, FPN, and YOLO to help develop various technologies that are commercially available today [2]. Although adaptation of such algorithms along with other object detection techniques such as image tiling improved the detection of smaller objects, the detection of 20×20 pixel or smaller objects is still an open problem [2].

Common approaches of small object detection in highly regarded object detection frameworks emphasize the detection of either relatively small objects in high-resolution images or actually small objects in a rather simple background [24]. An example of the former may aim to detect buildings and structures using satellite imagery. An example of the latter may aim to detect people on a clear sandy beach using drone images. Detection of such objects does not much suffer from the lack of spatial resolution, signal-to-noise ratio [3], and insufficient features inherent to the detection of objects smaller than 10×10 pixels and located in a complex background. Li et al. [4] adopt a perpetual GAN to enhance the low-level feature representation of small objects. Pham et al. [5] introduce YOLO-Fine as a one-stage small object detection method based on a much fine detection grid. The authors claim that their system is capable of detecting objects as small as 10×10 pixels. Zou et al. [6] and Tong et al. [7] present surveys of methods used in small object detection and object detection respectively. More recently, Liu et al. [8] present a comprehensive survey of research studies done in the field of small object detection. Some examples of deep learning techniques are fusing feature maps, adding context information, balancing foreground-background examples, and creating positive examples. Their experiments compare the popular models YOLOv3, Faster R-CNN, and SSD. Although the accuracy of the models is poor (< 40%), Faster R-CNN performed the best, while YOLOv3 was a close second. The limitations on size and complex backgrounds make these algorithms inadequate for detecting very small objects such as traffic signs or small figures of animals in their natural habitat. Another problem with the common small object detection deep learning algorithms is that they need to be trained on hundreds of larger images than the size of a small object being detected [9]. Therefore, training and inference of such algorithms demand a lot of computational power and memory [10].

In this study, we focus on the detection of objects in 5^2 – 8^2 pixel range that appear in complex and inconsistent background. Our initial in-person evaluations reveal that detecting such objects in an image could be next to impossible. Hence, even detecting these objects at a level would be very promising because, to the best of our knowledge, the traditional state-of-the-art methods could not delve into such smaller sizes. Detecting such small objects would enable making decisions in a timely manner for real-world applications.

We use 8×8 pixel bounding boxes to create Region of Interest (ROI) with a fixed coordinate stamp to help track the location of the detected objects. Our method primarily consists of three components: *unsupervised ROI extraction, image super-resolution, and two-stage cascade classifier*. Each trainable component is

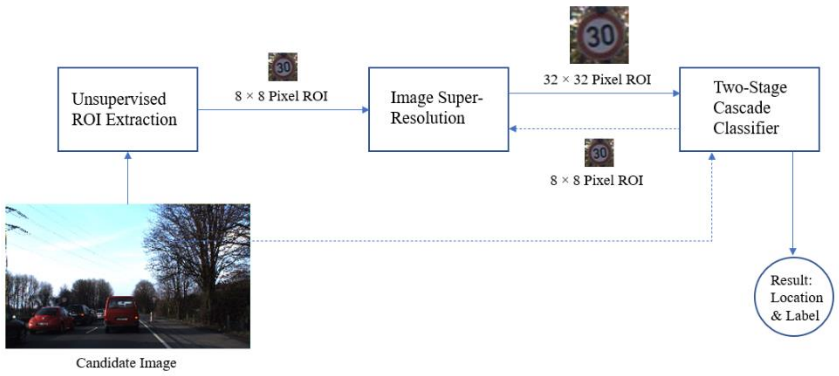

Figure 1: The architecture of the proposed framework: An input candidate image is used to extract ROIs. Then the ROIs are upsampled and sent for classification. The upsampled ROIs are first classified into the small object of interest (in this case traffic signs) or the background. Next, the ROIs with the object of interest are classified into respective classes.

separately trained with a maximum resolution of 32×32 pixels, which is only 16 – 40 times larger (in resolution) than the small object of interest. After completing training, each component is merged into a pipeline in the high-level architecture shown in Figure 1. An example of object detection is shown in Figure 2.

For experiments, we choose the publicly available Traffic Sign Recognition Benchmark dataset (GTSRB) [11] and the German Traffic Sign Detection Benchmark dataset (GTSDB) [12] for the following reasons. Detecting traffic signs could be challenging due to the complex background, various illuminations, different orientations, and the presence of hard negatives [13]. In addition, the inconsistency of the background of traffic signs helps avoid potential explainability pitfalls [14]. Moreover, when downsampling is applied to the GTSDB dataset, most traffic signs are hardly seen by the naked eye due to the lack of high-level features. Thus, the resulting dataset can be considered a challenging domain for any small object detection algorithm compared to the benchmark MS COCO [15] and DOTA [16] datasets.

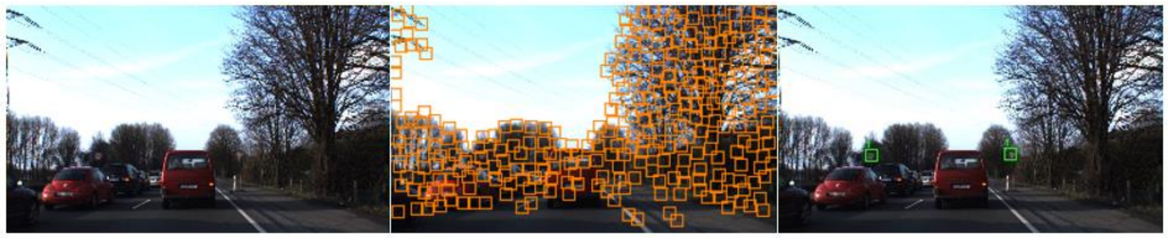

Figure 2: (Left) The input image. (Middle) The extracted ROIs. (Right) The output image with detected traffic signs.

2 METHODOLOGY

In this section, we explain the main components of the proposed framework including *unsupervised ROI extraction, image super-resolution*, and *two-stage cascade classifier* shown in Figure 1. The selection of hyper-parameters of these components is discussed in Section 3.

2.1 Unsupervised ROI Extraction

There are numerous techniques for automatic ROI extraction using general-purpose ROI extraction and domain-specific approaches. These methods can also be categorized as supervised or unsupervised techniques. Nevertheless, most of the popular automated state-of-the-art ROI extraction methods do not meet the criteria mentioned in the introduction since they are either supervised or focused on a specific feature of the target [11,17,18]. This led us to propose a novel unsupervised ROI extraction framework for small object detection composed of three major processes: *primary contour detection, selecting overlapping tiles, and random ROI generation*. Each process is unsupervised and executed sequentially.

2.1.1 Primary Contour Detection.

This is our novel approach that leverages global image thresholding for a range of threshold values T, where $T \in [l, h]$ and l and h are hyperparameters representing low and high pixel intensities respectively. In this paper, we choose multiples of 10 as the step factor from l to h and set l and h as 50 and 200 respectively. Then each binarized grayscale candidate image is used to produce a set of points belonging to small, disconnected contours having 2 – 10 pixels. This important task aims to limit the selected contours for small regions that could represent the features of small objects by applying connected component analysis. Finally, the union of the sets of points produced for all T is fed to the next stage, *selecting overlapping tiles*.

For very small object detection, the *primary contour detection* outperforms both adaptive image thresholding [19] and Otsu image thresholding [20]. Figure 3 shows a comparison between the adaptive image thresholding and the *primary contour detection* in the identification of small, disconnected contours. The adaptive image thresholding (left) proposes small contours from all parts of the image although there is a plane sky and there are smooth objects in the background whereas our *primary contour detection* (right) illustrates only edges and objects with a textured surface, and therefore, correctly guides the selection of tiles to extract ROIs. Although choosing a larger block size alleviates false positives (incorrect candidate regions) for adaptive image thresholding, it increases the risk of missing contours of the small traffic signs. Moreover, Otsu image thresholding faces a similar issue as adaptive image thresholding. All three methods are fast enough to avoid latency but only the *primary contour detection* produces satisfactory results leading to reliable and efficient ROI extraction.

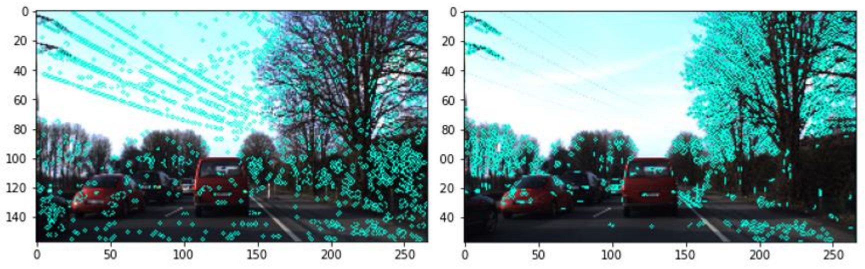

Figure 3: A comparison: (Left) Adaptive image thresholding highlights all regions of the candidate image. (Right) the *primary contour detection* highlights only regions with significant edges.

2.1.2 Selecting Overlapping Tiles

This is the second step of the proposed *unsupervised ROI extraction*, and it initially creates 32×32 pixel virtual tiles with an overlapping margin of 8 pixels. Therefore, any traffic sign slightly smaller than 8×8 pixels can be completely captured by at least one tile. If a point of small contour generated in the previous step intersects a virtual tile, this tile becomes a candidate slice of the region. Usually, this point-tile matching starts from the top-left corner of the candidate image and continues from left to right. To optimize this brute force comparison, we extract slices along with all matching points right after each pass from left to right. Thus, at the start of each pass, there is always a smaller number of points and tiles to compare.

2.1.3 Random ROI Generation

In the last step of the proposed unsupervised ROI extraction, in which the extracted 32×32 pixel slices are scanned with a multifilter procedure. Filter f_1 roughly locates the ROI. Filter f_2 is activated by a strong signal region whereas filter f_3 is designed to detect weak signals from a small object. The strength of a detected signal is measured by the total number of points belonging to any contour having several pixels in the range [n_1, n_2]. These values are a part of the hyperparameters of our model. We choose n_1 and n_2 to be 5 and 10 respectively in our experiments. First, an 8×8 pixel grid is applied to divide the slices into 16 smaller regions. Then each region is scanned in a sliding window fashion by filter f_1 using Canny edge detection with an automatic threshold estimation [21]. This process of region-based scanning aims to narrow down the search for ROI further by benefiting from the Canny edge detection to eliminate locally weak and noisy edges in our analysis.

After locating the regions with significant edges, at most N number of different bounding boxes are generated by shifting the approximate center of the square-shaped regions randomly in [-4, 4] neighborhood (along both axes). N is a hyperparameter of our model such that $1 \leq N \leq 64$ and is set as 20 in our experiments. For each bounding box, filters f_2 and f_3 are applied for scanning the overlapping regions. Filter f_2 is designed with a Gaussian filter and Canny edge detection (thresholds: 150, 400) to detect a strong signal from a small object. The purpose of filter f_3 is to detect a weak signal from a small object. This is achieved by implementing Canny edge detection with automatic threshold estimation to detect closed contours. To capture traffic signs failing to generate a strong signal, filter f_3 enables to detect a weak signal of a closed contour which is barely noticeable in the background. Finally, the ROIs extracted from all the slices are collected into a pool and standard non-maximal suppression (NMS) with an overlapping threshold of 0.1 is applied to select the best ROIs.

2.2 Image super-resolution

For the image super-resolution component of the proposed framework, we use the SRGAN [22], which is adapted from the notion of obtaining photo-realistic images using super-resolution. During the training, the SRGAN learns to upsample 8×8 pixel low-resolution images corresponding to 32×32 pixel high-resolution images of traffic signs. In our experiments, only the image patches of traffic signs are used to train the SRGAN from scratch. Despite each image including many background pixels, the model is trained with an intentional selection bias. Therefore, the pretrained generator is much more capable of enhancing features inherent to the traffic signs. We leverage this bias to produce more correct and detailed images of high-resolution traffic signs as generating images like the original traffic signs is important for recognition. The risk of developing features resembling to traffic signs in the background is mitigated using our two-stage cascade classifier.

2.3 Two-stage cascade classifier

The last component of our framework is the two-stage cascade classifier that consists of two ResNet50 [23] models. The architectures of these two classifiers are identical except for their output layer. Both classifiers are trained from scratch by using training sets of super-resolution images of ROIs. Figure 4 shows the two-stage cascade classifier as a single classification pipeline, which takes the super-resolution ROI as an input and outputs the predicted class of the traffic sign and the location. Stage 1 of the cascade classifier is a binary classifier that classifies the input ROI as a traffic sign or background. The prediction probability *p* of the desired class is partitioned into [0, *L*), [*L, H*), and [*H, L*], where *L* and *H* are hyperparameters representing low and high thresholds of *p*. Based on the results of this stage, the following classification and subsequent actions are applied on the ROI.

I. If $p \in [0, L)$, then the ROI is classified as the background and discarded.
II. If $p \in [L, H)$, then the ROI is classified as a potential traffic sign and sent to *optimizing ROI*.
III. If $p \in [H, 1]$, then the ROI is classified as a traffic sign and sent to stage 2.

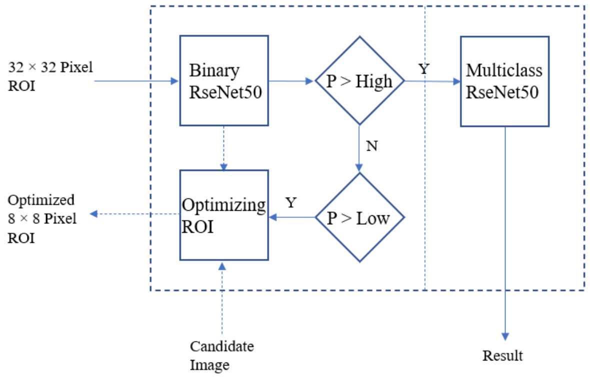

Figure 4: Two-stage cascade classifier: The main inference pipeline takes a super-resolution ROI as the input. In stage 1, it is classified into traffic signs or background. If the ROI is classified as a traffic sign, then it is sent to stage 2 in which it is classified into the respective classes.

We set *L* and *H* to be 0.01 and 0.95 for our experiments. The *optimizing ROI* is designed to address the partial presence of the small object in the super-resolution ROI. The idea is to replace such weak ROI with an optimized ROI that has a better presentation of the small object. Thus, this process can access the candidate and scan the neighborhood of the original ROI. It produces a series of optimized ROIs by shifting the location of the approximate center *c* of the original ROI to a new location within the original ROI.

Optimizing ROI uses an optimal path search presented in Algorithm 1 to determine the best-optimized ROI that yields the local maximum *p*. Let $G(V, E)$ be the graph representing the association between the neighboring pixels of the original ROI. Thus, *V* is the set of pixels and *E* is the set of associations between neighboring pixels of the original ROI. The following steps yield the best-optimized ROI to be sent to the main inference pipeline. If an optimized ROI fails to be classified as a traffic sign, it is considered to be the background and discarded.

In stage 2, a multiclass model classifies the ROIs containing the traffic signs into respective classes. For this task, a multiclass ResNet50 is trained to classify the traffic signs into 6 classes as proposed by the authors of the datasets [11]. We trained our binary classifier using 32×32 pixel super-resolution images of traffic signs and background with approximate ratio 1:4. By modeling a binary classifier to learn a wide range of features of the background, this provides the binary classifier an advantage to classify ROIs with background more correctly while offsetting a possible side effect of irrelevant features in the background of traffic signs due to non-square shapes of traffic signs and limitations of image super-resolution.

ALGORITHM 1: Optimizing ROI

```
//Perform an optimal path search on $G$.
//**Input**: $G(V, E)$, $c$
//**Output** optimized $c$
**while** $c \in V$ **do**

    Find the unprocessed neighborhood $W$ of $c$ ($W \subset V$)
    $k = \underset{v \in W}{\operatorname{argmax}}(p_v)$  //$p_v$ is $p$ of the ROI of which the center pixel is the node $v$
    **if** $p_k > p_c$ **then**
    set $c = k$
    **end**

**end**
```

3 DATASETS AND RESULTS

3.1 Data Preprocessing

We preprocess the original datasets GTSRB (traffic sign images) and GTSDB (road images having traffic signs) for our experiments as follows. For the training of the proposed framework, we construct the datasets known as *TV1*, *TV2*, and *TV3*. For the evaluation of the proposed framework, we construct a benchmark dataset known as *BMV*.

TV1: This dataset is created by randomly selecting 10,473 images from the GTSRB dataset and rescaling them to 32×32 resolution images. *TV1* is primarily created to train the SRGAN model. Images are downscaled for training the SRGAN. We use the split ratio of 0.8 for training and 0.2 for testing.

TV2: This dataset consists of all super-resolution images obtained from *TV1* and super-resolution images obtained from 39,527 low-resolution background patches extracted from selected 108 images of the GTSDB validation dataset produced by 4× downsampling. First 8×8 pixel patches are selected from the most probable regions of the downsampled images using our primary contour detection and then the extracted patches are upsampled using the trained SRGAN. Then a total of 50,000 images go through the following data augmentation: zoom range = 0.2, width shift range = 0.1, and height shift range = 0.1. Again, the split ratio is 0.8 for training and 0.2 for testing.

TV3: This dataset consists of super-resolution images of randomly selected 100 images from all classes of the GTSRB dataset. Thus, there are 4300 images in TV3. First, the images are downscaled to the resolution of 8×8 and then upsampled using the trained SRGAN. Then these images are combined to create 6 super classes suggested by the authors [15]: Class 1: speed limit signs, Class 2: other prohibitory signs, Class 3: derestriction signs, Class 4: mandatory signs, Class 5: danger signs, and Class 6: unique signs. The split ratio: 0.8 and 0.2.

BMV: We categorize the GTSDB dataset into 3 groups based on the scene: rural, suburban, and urban. Rural images contain a more natural environment and less traffic, whereas suburban images have some natural scenes and buildings with moderate traffic. On the other hand, urban images capture less natural environment, but they have more buildings and traffic. We select and annotate the remaining 182 images of the GTSDB validation dataset using a square-shaped bounding box where the area of all the images is at least 164 times larger than that of the bounding box. Then these images are scaled until the bounding boxes have the size of a 8×8 block. We define 3 different scene categories for the GTSDB dataset called rural, suburban, and urban. Rural: more natural environment and less traffic, Suburban: natural, built environment and traffic Urban: less natural environment, more built environment, and traffic. Then, 60 images are chosen using random stratified sampling from the above 182 images. The mean resolution of a *BMV* image is 190×322.

3.2 Analysis of Results

All our experiments were performed on Python 3 and TensorFlow 2.9.2 on Google Colaboratory with Tesla T4 GPU. We set the hyperparameters of our model as follows: l = 50, h = 200, N = 20, L = 0.01, H = 0.95. In our experiments, we have used Intersection Over Union (IOU) threshold of 0.3 for detection. Given a successful locating of traffic signs, the mean IOU value was 0.637 with a standard deviation of 0.182. The average number of traffic signs per image is 1.65. The average number of predictions made by the model per image is 3.32. We have obtained Average Precision (AP) of 0.332. Table 1 presents the precision and recall for each class based on the *BMV* dataset. Class 1 and Class 5 performed relatively better than other classes. However, the performance in Class 4 was the lowest among all classes. Figure 5 shows a sample of the results in which locating a traffic sign is presented with a bounding box along with the predicted class on top of it. Locating an object other than traffic signs is shown with an unlabeled bounding box.

Table 1: Detection of traffic signs by the proposed framework on the *BMV* dataset

Category	Number of Ground Truth	Number of Predictions	TP	FP	FN	Precision	Recall
Class 1	23	37	12	25	11	0.324	0.522
Class 2	14	44	5	39	9	0.114	0.357
Class 3	9	28	3	25	6	0.107	0.333
Class 4	15	25	3	22	12	0.12	0.2
Class 5	19	30	11	19	8	0.367	0.579
Class 6	19	35	7	28	12	0.2	0.368

3.3 Discussion

The hyperparameters of our framework need to be tuned for optimal performance. To determine thresholds l and h, we recommend using primary contour detection separately on a sample of the validation images and examine its performance on different l and h values. This step just requires one-time selection at the beginning of evaluation. Setting a high number N for the random bounding boxes to be processed by the unsupervised

ROI extraction completely on the sample image may result in a longer time to complete this task, but it may yield better ROI extraction. The last hyperparameters are for thresholds *L* and *H.* To determine these values, the proposed framework needs to be executed at least up to Stage 1 using the sample images

The proposed method is also a low-cost object detection framework. For efficient training of the proposed method, the SRGAN needs to be trained using only the patches with the small object of interest. Also, the binary classifier in stage 1 should be trained with significantly more background patches compared to the images of the small objects. Consequently, the binary classifier has high specificity. This effect helps mitigate the potential bias of the features of upsampled ROIs generated by the trained SRGAN. In addition, the background patches should be a representative sample of the background features of the candidate images. *TV2* is constructed mainly using the background patches of rural settings. It is an average representative of suburban backgrounds and a poor representative of urban backgrounds. As in Figure 5 below, the effect of this selection bias in the training of the binary classifier is prominent in the evaluation of the proposed method using the *BMV* dataset. In general, more FP is produced for the candidate images of urban settings than images of rural settings.

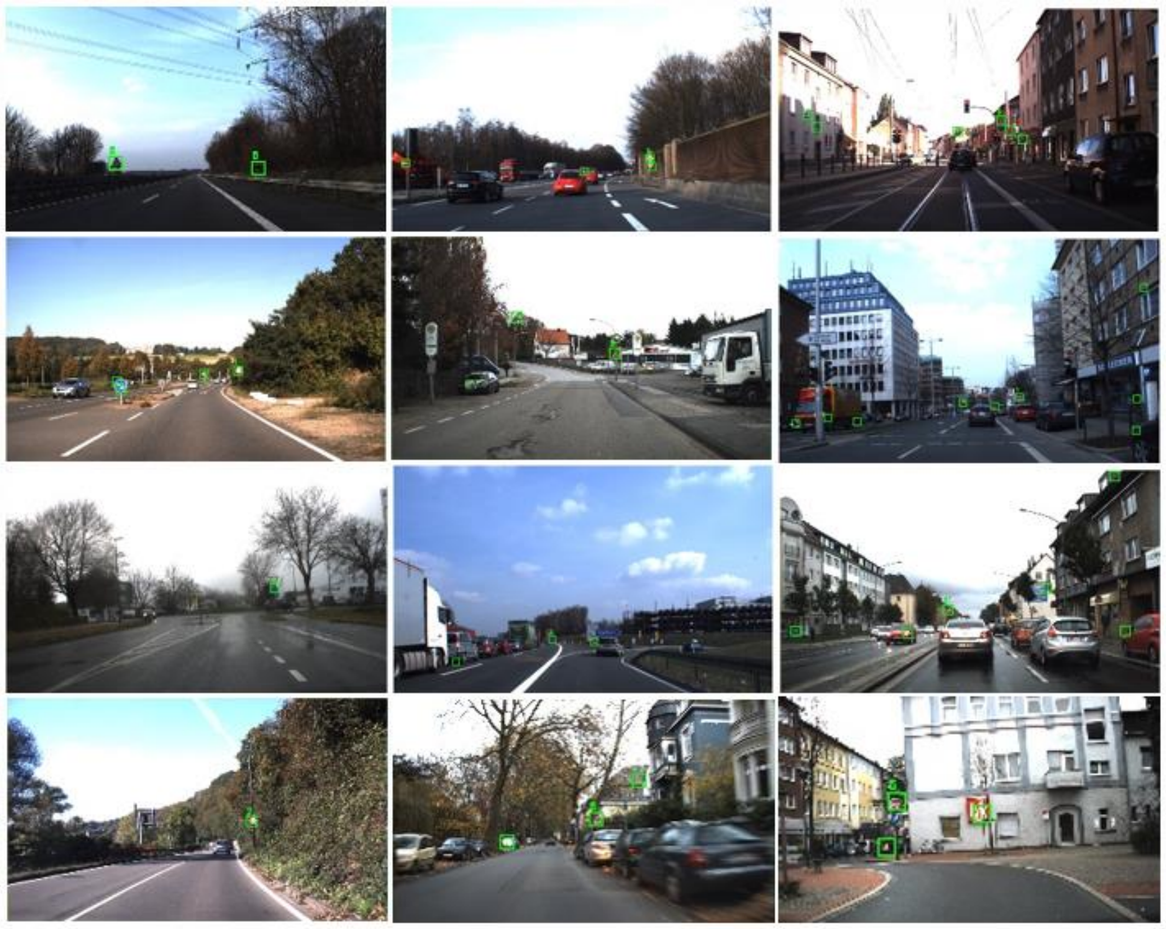

Figure 5: A sample of output images produced by the proposed framework using the *BMV* dataset. The localizations are presented with a 8×8 pixel bounding box along with the predicted class on top of it. FP is shown with a bounding box only. (Left) scene images of rural settings. (Middle) scene images of suburban settings, (Right) scene images of urban settings.

4 CONCLUSION

This research primarily investigates the possibility of developing a low-cost multistage framework to detect very small objects in a complex background. The major contribution of our work is to show the possibility of detecting very small objects as small as 5x5 pixels that appear in complex backgrounds, which has not been achieved before. While most small object detection algorithms are based on cutting-edge object detection models, our method combines a novel unsupervised ROI extraction with independently trained deep learning models that are not specific to object detection. The proposed framework successfully addresses the major challenges in small object detection: determination of location and class of small objects that has noisy low-level features. Our method demonstrates AP of 0.332 at IOU of 0.3 on an evaluation dataset with small traffic signs in a 5^2 – 8^2 pixel scale. The performance could be improved by using more data in each training phase is required, especially for training the binary classifier. In addition, a fast and more accurate image super-resolution technique would be beneficial. There are two major extensions planned for this work: reduce the running time and extend experiments to different datasets.

REFERENCES

[1] Fengqiang Xu et al. 2018. Real-time detecting method of marine small object with underwater robot vision. 2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO) (2018). DOI:http://dx.doi.org/10.1109/oceanskobe.2018.8558804

[2] Yang Liu, Peng Sun, Nickolas Wergeles, and Yi Shang. 2021. A survey and performance evaluation of deep learning methods for small object detection. Expert Systems with Applications 172 (2021), 114602. DOI:http://dx.doi.org/10.1016/j.eswa.2021.114602

[3] J.-C. Yoo and C.W. Ahn. 2012. Image matching using peak signal-to-noise ratio-based occlusion detection. IET Image Processing 6, 5 (2012), 483. DOI:http://dx.doi.org/10.1049/iet-ipr.2011.0025

[4] Jianan Li, Xiaodan Liang, Yunchao Wei, Tingfa Xu, Jiashi Feng, and Shuicheng Yan. 2017. Perceptual generative adversarial networks for small object detection. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017). DOI:http://dx.doi.org/10.1109/cvpr.2017.211

[5] Minh-Tan Pham, Luc Courtrai, Chloé Friguet, Sébastien Lefèvre, and Alexandre Baussard. 2020. Yolo-Fine: One-stage detector of small objects under various backgrounds in remote sensing images. Remote Sensing 12, 15 (2020), 2501. DOI:http://dx.doi.org/10.3390/rs12152501

[6] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. 2019. Object detection in 20 years: A survey. arXiv preprint arXiv:1905.05055 (2019).

[7] Kang Tong, Yiquan Wu, and Fei Zhou. 2020. Recent advances in small object detection based on Deep Learning: A Review. Image and Vision Computing 97 (2020), 103910. DOI:http://dx.doi.org/10.1016/j.imavis.2020.103910

[8] Yang Liu, Peng Sun, Nickolas Wergeles, and Yi Shang. 2021. A survey and performance evaluation of deep learning methods for small object detection. Expert Systems with Applications 172 (2021), 114602. DOI:http://dx.doi.org/10.1016/j.eswa.2021.114602

[9] Nhat-Duy Nguyen, Tien Do, Thanh Duc Ngo, and Duy-Dinh Le. 2020. An evaluation of deep learning methods for small object detection. Journal of Electrical and Computer Engineering 2020 (2020), 1–18. DOI:http://dx.doi.org/10.1155/2020/3189691

[10] Shrey Srivastava, Amit Vishvas Divekar, Chandu Anilkumar, Ishika Naik, Ved Kulkarni, and V. Pattabiraman. 2021. Comparative analysis of Deep Learning Image Detection Algorithms. Journal of Big Data 8, 1 (2021). DOI:http://dx.doi.org/10.1186/s40537-021-00434-w

[11] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. 2011. The German Traffic Sign Recognition Benchmark: A multi-class classification competition. The 2011 International Joint Conference on Neural Networks (2011). DOI:http://dx.doi.org/10.1109/ijcnn.2011.6033395

[12] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. 2013. Detection of traffic signs in real-world images: The german traffic sign detection benchmark. The 2013 International Joint Conference on Neural Networks (IJCNN) (2013). DOI:http://dx.doi.org/10.1109/ijcnn.2013.6706807

[13] Ming Liang, Mingyi Yuan, Xiaolin Hu, Jianmin Li, and Huaping Liu. 2013. Traffic sign detection by ROI extraction and histogram features-based recognition. The 2013 International Joint Conference on Neural Networks (IJCNN) (2013). DOI:http://dx.doi.org/10.1109/ijcnn.2013.6706810

[14] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "" Why should I trust you?" Explaining the predictions of any classifier." . Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2016). DOI:http://dx.doi.org/10.1145/2939672.2939778

[15] Tsung-Yi Lin et al. 2014. Microsoft Coco: Common Objects in Context. Computer Vision – ECCV 2014 (2014), 740–755. DOI:http://dx.doi.org/10.1007/978-3-319-10602-1_48

[16] Gui-Song Xia et al. 2018. Dota: A large-scale dataset for object detection in aerial images. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018). DOI:http://dx.doi.org/10.1109/cvpr.2018.00418

[17] Jiayuan Gu, Han Hu, Liwei Wang, Yichen Wei, and Jifeng Dai. 2018. Learning region features for object detection. Computer Vision – ECCV 2018 (2018), 392–406. DOI:http://dx.doi.org/10.1007/978-3-030-01258-8_24

[18] Huimin Lu, Yifan Wang, Ruoran Gao, Chengcheng Zhao, and Yang Li. 2021. A novel roi extraction method based on the characteristics of the original finger vein image. Sensors 21, 13 (2021), 4402. DOI:http://dx.doi.org/10.3390/s21134402

[19] A.D. Sappa. 2006. Unsupervised contour closure algorithm for range image edge-based segmentation. IEEE Transactions on Image Processing 15, 2 (2006), 377–384. DOI:http://dx.doi.org/10.1109/tip.2005.860612

[20] Jan Hosang, Rodrigo Benenson, and Bernt Schiele. 2017. Learning non-maximum suppression. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017). DOI:http://dx.doi.org/10.1109/cvpr.2017.685

[21] A. Rosebrock. 2015. Zero-Parameter. Automatic Canny Edge Detection with Python and OpenCV.[(accessed on 27 August 2021)] (2015).

[22] Christian Ledig et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017). DOI:http://dx.doi.org/10.1109/cvpr.2017.19

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016). DOI:http://dx.doi.org/10.1109/cvpr.2016.90

[24] Liu, Ying, et al. 2021. Survey of video based small target detection. Journal of Image and Graphics 9.4 (2021): 122-134.