Semiconductor Optoelectronics
ISSN: 1001-5868

A DEEP LEARNING SYSTEM FOR OBJECT DETECTION IN STREET IMAGES USING REGION- CONVOLUTIONAL NEURAL NETWORK

P. Ravija
M.Tech Scholar, Department of ECE, Marri Laxman Reddy Institute of Technology and Management (MLRITM), Dundigal, Medchal, Hyderabad-500043
Email:ravijachowdary96@gmail.com

Dr. K. Naveen Kumar
M.Tech.,Ph.D, Professor, Department of ECE, Marri Laxman Reddy Institute of Technology and Management (MLRITM), Dundigal, Medchal, Hyderabd-500043
Email: naveenkarunya@gmail.com

Dr. Srinivas Bachu
M.Tech., Ph.D, Department of ECE, Marri Laxman Reddy Institute of Technology and Management (MLRITM), Dundigal, Medchal, Hyderabd-500043,
Email: srinivasbachu@gmail.com

Abstract
Even while object identification methods have found widespread use in a variety of practical applications, automated tree detection is still a challenging problem to solve, particularly for street-view photos. The conventional machine learning networks would be used to recognise street objects automatically. Due to the obvious difficulties involved, low light and severe occlusion circumstances in tree detection have not been thoroughly researched up to this point. So, to resolving these challenges, this work describes a way of adjusting picture brightness that is both straightforward and capable of effectively addressing low-light conditions. In addition, this work proposes a new loss and a tree part-attention module for the purpose of lowering the number of erroneous detections brought on by strong occlusion. This work presented the occlusion-aware region- convolutional neural network (OAR-CNN) work, which is trained with multiple number of images. It is shown that the resultant framework, which is part attention network for tree recognition, can recognise objects in street-view photos in an effective manner. The simulation results show that, the proposed method resulted in higher performance as compared to the conventional approaches.

Keywords: Region- Convolutional Neural Network, Deep Learning, Objects detection, Objects recognition, Transfer learning.

1. Introduction

The tree has evolved into an element that cannot be overlooked in heavily populated cities. Most objects are situated along the roadways [1], where they play an essential part in the urban infrastructure of the city. They serve as multipurpose systems that reduce dust and noise, provide shade for pedestrians, and other similar functions. As a result, it is essential to track

their development and overall health [2]. Finding out how many of them there are the first and most critical step in this process. The government is interested in gathering information on the precise number of street objects present in a certain area as well as the kind of objects that make up those street objects. In the past, the only way to resolve this issue was by human labour. It was requested that knowledgeable individuals walk onto the road and count and categorise the objects that lined the road in sequential order, which clearly demanded a significant amount of time and manpower.

As more and more data are gained concerning the many advantages that urban forests provide to human health and well-being via a variety of ecosystem services, they are garnering increasing attention all over the world. During fast urbanisation trends [3], climate change, and expanding global commerce, planning, and maintaining urban forests and objects based on urban tree inventories is rapidly coming to the front. Regrettably, urban forest planning and management continues to be a formidable obstacle all over the world due to the relatively limited information that is available on the spatial distribution and accessibility of urban forests and objects, as well as their state of health, composition, structure, and function [4]. Most towns continue to conduct labour-intensive field surveys to gather and maintain inventories of public objects, but they lack information on private objects. Due to the high expense of mapping and monitoring objects over broad regions and over long periods of time, national and local sources of tree inventory are lacking in depth, consistency, and quantity [5]. This is even though urban objects are very important.

Many novel models and strategies have surfaced in recent times, as a direct result of the development of deep learning (DL) and the accessibility of enormous datasets. These innovative approaches have shown significant potential to bring about a sea change in the way certain challenges are approached [6]. The body of relevant literature continues to grow with the addition of research projects that make use of DL to produce superior outcomes. Computer vision (CV) [7] is an area of study that has made significant strides in the last few years, and a significant amount of research effort is now being focused into the development of CV applications. One of the fundamental building blocks of CV models is called CNN, and it outperforms many of its classic machine learning (ML) rivals. As a direct result of this, CV tasks like as object identification [8], object tracking, and semantic or instance segmentation have attracted a lot of attention. This area of study has progressed to the point where robust models and datasets are readily accessible to the public and may even be utilised for commercial and corporate applications, such as autonomous vehicles and healthcare [9]. Currently of digital technology, several urban research studies are attempting to include approaches that rapidly process and evaluate the complexity of urban dynamics and the rapid changes that are occurring all around them [10]. Learning algorithms based on machine learning and DL are effective instruments for the automation of complicated tasks for a variety of urban features. For instance, numerous studies use DL models to address urban issues, such as garbage management, urban environment quality, structure damage detection, and traffic prediction, as well as other issues. The novel contributions of this work

- Implementation of image Preprocessing method, which enhances the regions of input images and contrast improvement with noise removal.
- Development of OAR-CNN for detecting the objects from the input images, which can be used for real time applications.

Rest of the paper is organized as follows: Section 2 contains the detailed analysis of literature survey. Section 3 contains the detailed analysis of proposed methodology. Section 4 contains the detailed analysis of simulation results.

2. Literature survey

In [11] authors built a CNN model based on the famous You Only Look Once (YOLO) network and utilized a MobileNet as the backbone for feature extraction to recognize objects and their separate portions in Google Street View photos. The backbone technology that made this possible was a MobileNet. In [12] authors exactly estimated of the percentage of picture coverage provided by street objects. The studies performed are based on the researchers' own dataset. The test findings demonstrate that street objects can be identified with a success rate of 94.91%, and that the ratio of street objects to roadway is 16.30% and 13.81% in the two metropolitan environments utilized for testing. This may be observed in the study of [13], which exhibits street-level imagery, which may be utilized to fine-tune and deploy tailored item detectors in urban environments. There are a total of 763 pictures here, each of which has been labeled with a bounding box indicating which of five different landscape aspects it depicts.

In [14] authors adopted a technique of individual tree segmentation called Shadow-cut to determine the outlines of the street objects from the point cloud. Using a binary classifier, they first cut the forest into sections (like a support vector machine). This classification scheme is based on the geometric properties of the point cloud. Authors were able to solve [15] in 570 ms with an accuracy of 82.24 percent. It's true that Shripad Bhatlawande and company are responsible for this. The decision tree classifier is the optimal method for recognizing objects. Twenty tests were done to confirm its reliability. In [16] authors developed the RBDR mechanism, which has been included into a harvesting robot platform, and it has been tested in a lab emulating an orchard and in an outdoor environment with actual pomegranate objects. In this display, we provide the results of various studies. In [17] authors developed a multiclass variant of the O2PF, an Optimum-Path Forest-based oversampling algorithm, to generate synthetic samples using attributes extracted from photos of five urban tree species.

Using a grid index and other local parameters, in [18] authors developed a pointwise approach for detecting street treetops from MLS point clouds. Using the grid index of the MLS point clouds, a two-level neighborhood search approach is given to begin retrieving the region around a single point. This objective may be accomplished by using the strategy. An improved approach for determining tree height from tree fisheye photographs was developed by authors [19]. To extract the tree height extreme points shown in fisheye pictures, we offer YOLOX-tiny, a refined lightweight target recognition network. In [20] authors created a better approach for extracting street objects from MLS point cloud data. The group's goal was to isolate one street tree's point clouds from the chaos of a city roadway. The first step is to devise a means of identifying the pole-like elements. In [21] authors used state-of-the-art methods from DL for object detection to develop a single, fully trainable network for automated street tree recognition. This system provides the infrastructure for automatic street-object identification. Because to the difficulties inherent in such conditions, the literature has paid little attention to tree recognition situations in low light or with severe occlusion until now.

According to a method developed by authors [22], tree inventories may be created rapidly and affordably in any metropolitan area with sufficient publically accessible street-level pictures.

What makes our method unique is that it makes use of a Mask. The objects in the street footage were identified and localized with the use of a Regional CNN (Mask-CNN). Given the lack of research into innovative approaches for single tree edge detection in remote sensing applications. In [23] authors conducted an evaluation of these methods. Twenty-one cutting-edge deep-learning algorithms, including anchor-based (both single- and multi-stage) and anchor-free methods. In [24] authors focused on structural measurement, which was performed by reconstructing the 3D images using a photogrammetric method called structure from motion. In addition, software was unveiled for doing automated tree detection in 3D images. Using a supervised learning method, authors were able to train detectors [25] that utilised tree edge and trunk detectors but worked on points directly rather than a collection of voxels acquired through spatial division. Using the grid index, we were able to extract sixteen local statistical parameters from the sphere domain for each grid point.

3.Proposed Methodology

In this work, we introduce OAR-CNN platform that automatically detects and extracts urban elements in cities by making use of sophisticated DL and computer vision algorithms. The OAR-CNN is a fully equipped platform whose primary function is to analyse GPS and video data of a certain area to provide visualisations and data on a wide variety of urban characteristics and objects. Some of the elements that may be identified and localised by the platform include things like objects, garbage and recycling bins, street light poles and lamps, automobiles, traffic signals, and traffic signs, as well as other features like the width of the road or the availability of sidewalks. To accomplish this goal, OAR-CNN utilises a wide variety of cutting-edge computer vision techniques, including object identification, semantic segmentation, multiple item tracking, depth estimation, and many more. In the long run, the goal of OAR-CNN is to develop an extendable platform that can automatically record and extract many visual characteristics from metropolitan streets, therefore gradually doing away with the need for visual inspection. Urban planning administrators and policymakers, municipal offices, geoinformatics firms, and other commercial organisations may all make use of the OAR-CNN online application, which offers an environment that is well-organized and simple to use. OAR-CNN takes street data as its input and outputs geotagged feature information, which may then be employed in accordance with the user's specific requirements. Figure 1 shows the proposed block diagram for object detection using OAR-CNN.

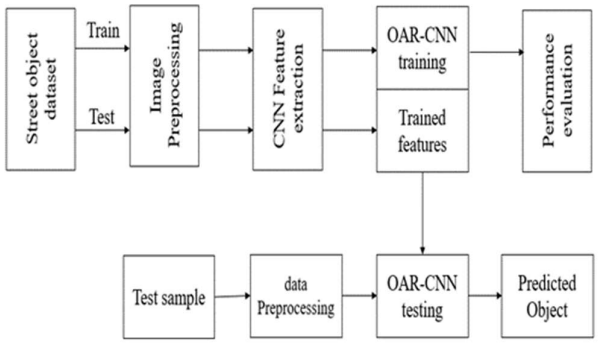

Figure 1. Proposed object detection using OAR-CNN.

The OAR-CNN is a deep learning system that is designed for object detection in street images. This system is able to detect objects even when they are partially occluded or hidden by other

objects in the scene. The OAR-CNN system is composed of several stages. The first stage is the input stage, where street images are fed into the system. The second stage is the feature extraction stage, where convolutional neural networks are used to extract features from the input images. The third stage is the region proposal stage, where the system proposes candidate regions in the image that may contain objects. The fourth stage is the occlusion-aware region pooling stage, where the system pools feature from the candidate regions to create a compact representation of the image. Finally, the fifth stage is the object detection stage, where the system uses a classifier to detect objects in the image. The classifier is trained using a large dataset of annotated street images and can accurately detect objects even when they are partially occluded or hidden by other objects in the scene. Overall, the OAR-CNN system is a powerful deep learning tool for object detection in street images and has many potential applications in areas such as autonomous driving, surveillance, and urban planning.

3.1 Image Preprocessing

Initially, relative point heights were adjusted to consider the terrain and low vegetation points that might have an impact on object identification. In this manner, action was taken. We disposed of anything that was less than a meter in height next. If the smallest tree in the sample is 1.5 meters in height, then eliminating data points from the edge data will not affect the morphologies of the edges. Local maxima points obtained in the following three iterations of the Tree OAR-CNN algorithm were utilized as the basis for comparison. To begin, a CHM with a resolution of 0.5 x 0.5 meters was constructed from the point cloud of the plot. Afterwards, a CHM-closing filter was built using a 3-by-3-pixel sliding window. Finally, a local maxima filter with a sliding window of size 3 by 3 pixels was used to extract the local maximum. If the elevation value in a local maxima pixel is more than 5 meters, that elevation is the local maxima point. The OAR-CNN took as input each local maximum value. For object detection algorithms to work, it is important to have bounding boxes of objects based on ground truth data. The dataset does not contain data on the edge sizes of the objects; therefore, it is impossible to build precise bounding boxes. The proximity of the objects makes it difficult to measure their edge diameters by hand. With this in mind, we collected information on the girths and heights of well-known objects with distinctive edges to formulate the regression equation. The regression equation allows us to calculate the edge sizes of all objects, from which we may derive accurate bounding boxes. A coordinate system for a three-dimensional bounding box may often be parameterized using the box's centre point coordinates together with the box's length, width, and height. As object roots tend to be low to the ground, we made the bottoms of the bounding boxes invisible. The lengths and widths of the enclosing boxes were used to represent the heights of the objects' edges since it was assumed that they were square at their bases. Therefore, the coordinate of a 3D bounding box was parameterized as (x*, y*, h*/2, w*, w*, h*) with four parameters. (x*, y*, h*/2) was the centre point coordinates. Here, W* was the edge size. H* was the tree height. The bounding boxes of reference objects were used as the ground truth bounding boxes for OAR-CNN to approximately determine the scopes of objects.

3.2. Feature Extractor

It was opted to conduct the feature extractor's pre-training, as shown in Figure 2. The CNN and the 4-layer network's starting parameters were discovered to be modifiable to improve accuracy, when utilizing the CNN detector. So, that it may be used by the subsequent extractor,

which identification was not a factor in the development of the weight and bias values; rather, they were derived via ImageNet picture training. Thereafter, the transferred network was fine-tuned by optimizing the initial feature extractor settings through the solution of an image classification issue, as will be detailed in more depth. It was hypothesized that the accuracy of object recognition might be improved by pre-training a four-layer neural network to discover the most important features of objects. Parameters of the 4-layer network were adjusted to solve the picture classification issue. The spherical data, which captures a full 360 degrees around its subject, was used to obtain both the object-filled and object less shots. Neither the learning nor the testing stages of the CNN detector used any of these. Images were classified using a binary system, with tree pictures being separated from other photos. Among the 400 images used for training, only 100 were randomly chosen for the tests. The proportion of pupils in each category was quite similar. To zero in on the optimal settings for training, sixty validation images were created that were distinct from the training and test images. These verification pictures were used throughout the training. Each set of three repetitions ended with a check to verify how well we have classified the data. Premature termination of training was implemented when the validation loss was more than or equal to the prior lowest loss for three iterations in a row. The CNN is the 4-layer network were both trained using a training set that included both object and non-object pictures, which demonstrates how the feature extractor was supported by the created network. Hence, modifications were made to CNNs weight and bias settings to address this issue.

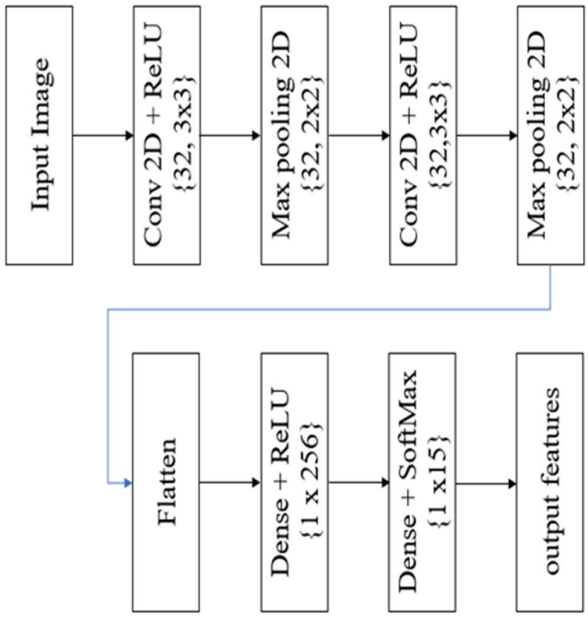

Figure 2. CNN feature extraction.

3.3 OAR-CNN

In the field of DL, researchers are still trying to figure out the optimal architecture for the CNN. There are several things to remember while making example changes. There are three components: the sample size, the sample patch size, and the image layers. In the current study, we made 8,000 sample patches to represent various types of things like objects. The size of the sample was determined to be exactly 22 × 22 pixels. Methods such as trial and error were

utilized with other approaches to find the best size of the sample patches. Under 22 × 22, the accuracy of the tree canopy recognition declined, while beyond 22 × 22, some of the smaller objects were left out. The great majority of the study area's small objects occupied a space of only 22 × 22 pixels, the researchers discovered. The CNN models need even numbers of bytes for their input training images. It follows that maximal pooling might be used throughout the model's development. Samples were created using the NDVI and CHM criteria. Tiff files with the produced sample patches were stored (Figure 8). Developing customized example patches for each class took around five minutes.

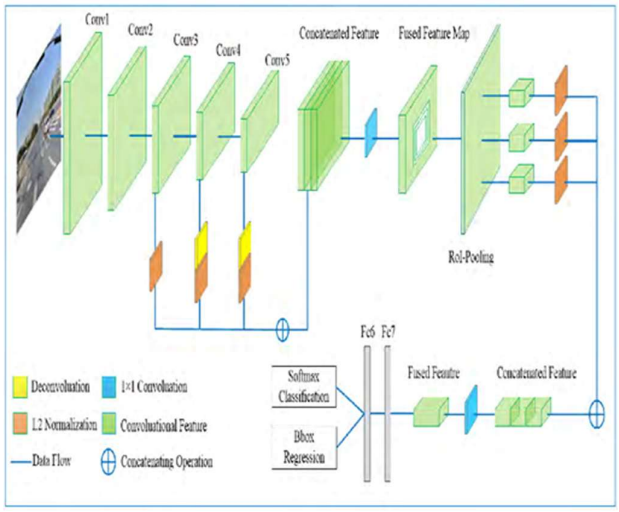

Figure 3. Proposed OAR-CNN model.

The processing time grows linearly with the required number of sample patches. Sample generation time will scale linearly with the quantity of samples requested. Each of the four spectral bands (green, red, infrared, and blue) was employed in the 2005 sample manufacturing from the respective photos. Nonetheless, merely the blue, green, and red channels were employed to produce samples from the 2015/2016 photographs. Figure 3 shows the architecture of proposed OAR-CNN.

Create OAR-CNN Model: It was decided to create a single-hidden-layer CNN model. The hidden layer is set according to the kernel size, the number of feature mappings, and the maximum pooling. The previous size kernels (13 x 13) were each given 40 feature maps because even-sized kernels would generate hidden units that are positioned between pixels, which would then need to be shifted to match the pixel bounds. Using max pooling with a 2 by 2 filter with a stride of 2 in both directions, we were able to reduce the feature map resolution. Both directions were covered by this action. Hence, the secret layer's nucleus equals the product of four times thirteen times thirteen times forty. The first component represents the number of image layers, while the second and third components define the number of units in the immediate area from which connections are transferred into the obfuscated layer (13x13). At the end, the number of feature maps generated is what is represented by the constant number 40. Hence, there are 27,040 unique weights in the hidden layer of this network, obtained by multiplying 13 by 13 by 40.

Train OAR-CNN Model: The tagged sample patches were then used to train the model using backpropagation, with the revised model weights serving as output. An important factor in statistical gradient descent optimization is the learning rate, which controls the size of the

weight change introduced at each iteration. It was via trial and error that the 0.0015 learning rate was settled upon. Training will go more slowly with smaller values and may become trapped in local minima, producing weights that are far from ideal. Increases in the learning rate will accelerate the training process but may not cause it to achieve its optimum minimum. Training iterations were set at 5000, and training samples were limited to 50. Even with the tagged samples and weight parameters supplied, it still took about 30 minutes to complete the training operation.

Apply OAR-CNN Model: After running the trained CNN model on the input picture (which had four layers in the 2005 image and three layers in the 2015 image), heatmaps for the object class were created (Figure 9). Apply CNN was the technique used by the recognition software. In the heatmaps, the numbers from 1 to 0 represent the interval in which a certain tree is most likely to be located (the values close to 1 indicate the high likelihood of objects and those close to 0 indicate a low likelihood of objects). Then, a 7 × 7 gaussian filter with a 32-bit float output type was used to refine the generated heatmaps. The goal was to remove the objects in the background. Using a morphological (dilate) filter with dimensions of three pixels on both the horizontal and vertical axes, we were able to create the local maxima of the smoothed heatmap of the objects.

Object-Based Classification Refinement: The heatmaps were segmented using multiresolution analysis with the parameters scale factor 10, shape 0.1, and compactness 0.5. Every tree probability larger than 0.5 belongs to the specialized tree category. The classification used an object threshold of less than or equal to 2 meters and an object criterion of less than 0.1 to reduce noise, since objects, grass, and non-tree objects have comparable spectral signatures. The criteria used led to a significant decrease in ambient noise. The arranged and improved tree objects were then refined using features like assign merge, pixel-based object scaling, and delete object. If a node in a tree was a neighbour of another node and their relational boundaries were larger than or equal to zero, then the two nodes were merged. Using surface tension values greater than or equal to 0.5 and box sizes in X, Y, and Z of 5, 5, and 1, successive growing and shrinking modes were applied using the pixel-based object resizing approach. The goal was to give the tree's nodes more distinct shapes. Using a set of pixel thresholds, we were able to exclude the finer pieces that did not seem like objects. As a result, no sub-200-pixel-wide pieces were included in the objects class.

4.Results

This section gives the detailed analysis of object detection performance, in real time applications. The publicly available dataset is used to evaluate the experiments. Various performance measures are measured and compared with other approaches.

4.1 Dataset

The COCO (Common Objects in Context) dataset is one of the most widely used datasets for object detection and segmentation tasks. It was created by Microsoft in collaboration with Carnegie Mellon University and is currently maintained by the COCO Consortium. The dataset contains over 330,000 images taken from natural scenes and includes a wide variety of object categories such as people, animals, vehicles, and household objects. The images were collected from various sources, including Flickr and Google Images, and were manually annotated with object bounding boxes and segmentation masks by a team of human annotators. The COCO dataset is organized into three subsets: train, validation, and test. The train subset contains over

82,000 images with 330,000 annotated objects, while the validation subset contains 40,000 images with 200,000 annotated objects. The test subset is released without annotations and is used for benchmarking purposes.

Each object instance in the COCO dataset is annotated with a bounding box that tightly encloses the object, as well as a segmentation mask that precisely outlines the object's shape. In addition, each object is assigned a category label from a list of 80 object categories, including person, car, and airplane. The COCO dataset also includes several additional annotations, such as human key point annotations for people, captions for each image, and relationships between object instances. The COCO dataset has been used to train and evaluate many state-of-the-art object detection and segmentation models and has been instrumental in advancing the field of computer vision. It continues to be a popular choice for researchers and practitioners working on object recognition and scene understanding tasks.

4.2 Results and Discussion

Figure 3 shows the example traffic sign detected outcomes using OAR-CNN. The OAR-CNN system can also be used for traffic sign detection in street images. In this case, the system would be trained using a dataset of annotated traffic sign images. During the feature extraction stage, the system would learn to identify features that are important for detecting traffic signs, such as color, shape, and texture. The region proposal stage would then propose candidate regions in the image that may contain traffic signs. The occlusion-aware region pooling stage would then pool features from the candidate regions to create a compact representation of the image that captures the important information for traffic sign detection. Finally, the object detection stage would use a classifier to detect traffic signs in the image. One advantage of using the OAR-CNN system for traffic sign detection is that it can handle partially occluded or hidden signs, which can be important in real-world scenarios where traffic signs may be partially obscured by other objects.

(b)

Figure 3. Example traffic sign detected outcomes using OAR-CNN.

Table 1 compares the performance of various object detection methods. The first column contains a performance metrics. The second column contains the performance estimation during the YOLO [11] method. The third column contains the performance estimation during the YOLOX-tiny [19] method. In the fourth column, the performance estimation during MASK-CNN [21] is presented. Finally, the last column contains the performance of proposed OAR-CNN of with all modules presented.

Table 1. Performance estimation of various object detection methods.

Metric	YOLO [11]	YOLOX-tiny [19]	MASK-CNN [21]	Proposed OAR-CNN
Accuracy (%)	92.82	94.75	96.47	99.28

Precision (%)	93.41	94.02	96.00	99.56
Recall (%)	93.38	94.16	96.97	99.02
F1-score (%)	93.79	95.35	96.10	98.91
AUC (%)	92.06	95.74	96.33	99.17

Table 2 shows the percentage of improvements from Table 1. Here, the proposed OAR-CNN method has increased accuracy by 7.16%, precision by 6.83%, recall by 6.30%, f1-score by 5.21%, and AUC by 7.96% as compared to the YOLO [11]. Then, the proposed OAR-CNN method has increased accuracy by 4.96%, precision by 5.82%, recall by 5.05%, f1-score by 2.30%, and AUC by 4.05% as compared to the YOLOX-tiny [19]. Finally, the proposed OAR-CNN method has increased accuracy by 3.01%, precision by 3.18%, recall by 2.49%, f1-score by 1.46%, and AUC by 2.93% as compared to MASK-CNN [21].

Table 2. Percentage of improvements of Table 1.

Metric	YOLO [11]	YOLOX-tiny [19]	MASK-CNN [21]
Accuracy (%)	6.95	4.78	2.91
Precision (%)	6.58	5.89	3.70
Recall (%)	6.03	5.16	2.11
F1-score (%)	5.45	3.73	2.92
AUC (%)	7.72	3.58	2.94

5. Conclusion

The OAR-CNN system can be trained and evaluated using the COCO (Common Objects in Context) dataset, which is one of the largest and most comprehensive datasets available for object detection tasks. The COCO dataset includes over 330,000 images with annotations for 80 object categories and has been widely used for benchmarking object detection systems. Training the OAR-CNN system on the COCO dataset can enable it to accurately detect and classify objects in street scenes, even in challenging scenarios with occlusions and cluttered backgrounds. By leveraging the occlusion-aware region pooling technique, the OAR-CNN system can effectively integrate information from multiple regions of an image to improve the accuracy and robustness of object detection. In conclusion, the combination of the OAR-CNN system and the COCO dataset can provide a powerful tool for object detection in natural scenes, with many potential applications in fields such as autonomous driving, surveillance, and robotics. Finally, the proposed OAR-CNN method has increased accuracy by 3.01%, precision by 3.18%, recall by 2.49%, f1-score by 1.46%, and AUC by 2.93% as compared to existing methods. This work can be extended with other transfer learning models for improved performance.

References

[1]. Ghasemi, Yalda, et al. "Deep learning-based object detection in augmented reality: A systematic review." Computers in Industry 139 (2022): 103661.

[2]. Choi, Ji Dong, and Min Young Kim. "A sensor fusion system with thermal infrared

camera and LiDAR for autonomous vehicles and deep learning based object detection." ICT Express (2022).
[3]. Hardalaç, Fırat, et al. "Fracture detection in wrist X-ray images using deep learning-based object detection models." Sensors 22.3 (2022): 1285.
[4]. Wang, Ning, Yuanyuan Wang, and Meng Joo Er. "Review on deep learning techniques for marine object recognition: Architectures and algorithms." Control Engineering Practice 118 (2022): 104458.
[5]. Zaidi, Syed Sahil Abbas, et al. "A survey of modern deep learning based object detection models." Digital Signal Processing (2022): 103514.
[6]. Córdova, M., Pinto, A., Hellevik, C. C., Alaliyat, S. A. A., Hameed, I. A., Pedrini, H., & Torres, R. D. S. (2022). Litter detection with deep learning: A comparative study. Sensors, 22(2), 548.
[7]. Mani, V. R. S., A. Saravanaselvan, and N. Arumugam. "Performance comparison of CNN, QNN and BNN deep neural networks for real-time object detection using ZYNQ FPGA node." Microelectronics Journal 119 (2022): 105319.
[8]. Sharma, Teena, et al. "Deep learning-based object detection and scene perception under bad weather conditions." Electronics 11.4 (2022): 563.
[9]. Jebamikyous, Hrag-Harout, and Rasha Kashef. "Autonomous vehicles perception (avp) using deep learning: Modeling, assessment, and challenges." IEEE Access 10 (2022): 10523-10535.
[10]. Hacıefendioğlu, Kemal, and Hasan Basri Başağa. "Concrete road crack detection using deep learning-based faster R-CNN method." Iranian Journal of Science and Technology, Transactions of Civil Engineering (2022): 1-13.
[11]. Ju, Yuanzhen, et al. "Loess landslide detection using object detection algorithms in northwest China." Remote Sensing 14.5 (2022): 1182.
[12]. Ahn, Hyochang, and Han-Jin Cho. "Research of multi-object detection and tracking using machine learning based on knowledge for video surveillance system." Personal and Ubiquitous Computing (2022): 1-10.
[13]. Roy, Arunabha M., Rikhi Bose, and Jayabrata Bhaduri. "A fast accurate fine-grain object detection model based on YOLOv4 deep neural network." Neural Computing and Applications (2022): 1-27.
[14]. Ghorbanzadeh, Omid, et al. "Landslide detection using deep learning and object-based image analysis." Landslides 19.4 (2022): 929-939.
[15]. Diwan, Tausif, G. Anirudh, and Jitendra V. Tembhurne. "Object detection using YOLO: Challenges, architectural successors, datasets and applications." Multimedia Tools and Applications (2022): 1-33.
[16]. Francies, Mariam L., Mohamed M. Ata, and Mohamed A. Mohamed. "A robust multiclass 3D object recognition based on modern YOLO deep learning algorithms." Concurrency and Computation: Practice and Experience 34.1 (2022): e6517.
[17]. Zhang, Yanchao, et al. "Real-time strawberry detection using deep neural networks on embedded system (rtsd-net): An edge AI application." Computers and Electronics in Agriculture 192 (2022): 106586.
[18]. Balasubramaniam, Abhishek, and Sudeep Pasricha. "Object detection in autonomous vehicles: Status and open challenges." arXiv preprint arXiv:2201.07706 (2022).

[19]. Yue, Xuebin, et al. "YOLO-GD: a deep learning-based object detection algorithm for empty-dish recycling robots." Machines 10.5 (2022): 294.
[20]. Zhu, Yanzhao, and Wei Qi Yan. "Traffic sign recognition based on deep learning." Multimedia Tools and Applications 81.13 (2022): 17779-17791.
[21]. Horváth, Dániel, et al. "Object detection using sim2real domain randomization for robotic applications." IEEE Transactions on Robotics (2022).
[22]. Zhou, Zhengxue, et al. "Learning-based object detection and localization for a mobile robot manipulator in SME production." Robotics and Computer-Integrated Manufacturing 73 (2022): 102229.
[23]. Obaidat, Islam, et al. "Jadeite: A novel image-behavior-based approach for java malware detection using deep learning." Computers & Security 113 (2022): 102547.
[24]. Tabassum, Fahima, et al. "Human face recognition with combination of DWT and machine learning." Journal of King Saud University-Computer and Information Sciences 34.3 (2022): 546-556.
[25]. Bjerge, Kim, Hjalte MR Mann, and Toke Thomas Høye. "Real-time insect tracking and monitoring with computer vision and deep learning." Remote Sensing in Ecology and Conservation 8.3 (2022): 315-327.