A Methodology for Automated Conversion of Axis-Aligned to Polygonal and Oriented Bounding Box Annotations in Aerial Imagery Object Detection

Daniela L. Freire[1](✉), Andre C. P. L. F.de Carvalho[1], Augusto José Peterlevitz[2], Mateus Antonio Chinelatto[2], Ricardo Dutra da Silva[2], and Juan Fernando Rojas Perea[3]

[1] Institute of Mathematics and Computer Sciences, University of Sao Paulo, Sao Carlos, SP, Brazil
{danielalfreire,andre}@icmc.usp.br

[2] Instituto de Pesquisas Eldorado, Campinas, Brazil
{augusto.peterlevitz,mateus.chinelatto,ricardo.silva}@eldorado.org.br

[3] Transmissora Aliança de Energia Elétrica S.A., Belo Horizonte, Brazil
juan.perea@taesa.com.br

Abstract. Object detection in aerial imagery presents significant challenges in computer vision due to the varied orientations and complex backgrounds of objects such as buildings and vehicles. Current annotation tools often fail to accurately delineate these objects, relying on manual bounding box methods that are both time-consuming and inconsistent.

Our novel methodology automates the conversion of axis-aligned annotations into polygonal and rotated annotations, prioritising systematic and scalable enhancements to data quality rather than modifying the model itself. Precise annotations, crucial for determining object locations and boundaries, are fundamental to this approach. We evaluated this methodology through a case study involving electrical transmission towers in aerial images, using advanced object detectors based on variations of the YOLOv8 algorithm. Preliminary results indicate that our automated method not only improves annotation accuracy but also significantly reduces the manual effort required, thereby lowering overall costs and time for data preparation in object detection training. The success of this methodology underscores its potential for broader applications and further advancements in automated annotation technologies.

Keywords: Object Detection · Image Segmentation · Machine Learning · YOLO Model · Segment Anything Model · Oriented Bounding Box · Polygonal Annotation · Annotation Conversion

1 Introduction

Artificial intelligence (AI), with a focus on data-centric approaches, has become a critical area of innovation, particularly in computer vision [27,30]. Correct and

V. Julian et al. (Eds.): IDEAL 2024, LNCS 15347, pp. 373–385, 2025.
https://doi.org/10.1007/978-3-031-77738-7_31

consistent annotations are essential, as they directly influence the effectiveness of training and the performance of object detection models [6]. Object detection in aerial imagery poses distinct challenges in the field of computer vision, mainly due to the diverse orientations and intricate backgrounds of objects such as buildings and vehicles [2,11].

Manual bounding box methods, the primary means of annotating images for object detection, pose significant challenges [5,28]. These challenges are particularly pronounced in aerial imagery, where objects such as vehicles or buildings often appear at various angles, complicating the annotation task [1,12,17]. Current annotation tools often lack support for oriented bounding boxes (OBBs) and polygonal annotations, which are crucial for accurately delineating inclined objects. Moreover, manually drawing bounding boxes is time-consuming and often inconsistent, leading to significant variations in annotation quality [26].

The laborious nature of manual annotations not only incurs high time costs but also impacts the quality of the resulting models. Inaccurate annotations can lead to poorly trained models, which may fail to detect objects correctly, compromising the reliability and effectiveness of AI applications in critical fields such as surveillance and remote sensing [4,14].

This article introduces an automated methodology to transform axis-aligned annotations into polygonal and rotated annotations compatible with advanced object detection and segmentation models such as YOLOv8, using the Segment Anything Model (SAM). YOLOv8 has been developed to address various object detection and segmentation challenges, including oriented bounding boxes. The automated conversion method aims to significantly reduce the time and cost associated with manual annotation processes by using these AI models. SAM, combined with YOLO models, is expected to improve annotation accuracy and efficiency, offering a reliable solution to the difficulties associated with manual annotation techniques.

We applied this methodology in a real case study involving electrical transmission towers in aerial images. These images introduce a unique set of complexities for computer vision researchers, especially in object detection and instance segmentation tasks. Transmission towers exhibit various configurations and are composed of linear elements with varying degrees of sparsity. Additionally, such images are marked by significant variations in background scenery, lighting conditions, and the relative sizes of objects across different captures. This case study not only tests the effectiveness of our proposed method in a challenging real-world scenario but also demonstrates the potential for broader application in similar settings.

The structure of this paper is as follows. Section 2 provides an overview of the background and a concise description of the methods and techniques used. The methodology to convert annotations is detailed in Sect. 3. The experimental study is discussed and reported in Sect. 4. The paper concludes with Sect. 6, where the principal findings are summarised and potential avenues for future research are explored.

2 Background

This section explains the different annotation methods used in various computer vision tasks such as object detection, instance segmentation, and oriented bounding boxes. Object detection involves identifying objects and their locations within images or videos. Instance segmentation goes further by precisely outlining each object to separate it from the background. Oriented bounding boxes extend regular object detection by adding an orientation aspect to improve the accuracy of locating objects in images.

Despite rapid progress in artificial intelligence, computer vision faces challenges in achieving the accuracy of human perception. In this context, the quality of the training data is as crucial as the algorithms themselves. The precision of the data annotations directly influences the effectiveness of model predictions. Various approaches to address this issue exist, each contingent upon the specific requirements of the use case, such as bounding boxes or polygons. Functionally, both approaches significantly simplify the process for algorithms to locate and identify objects within an image, linking detection to the entities on which they were initially trained.

Bounding boxes are rectangular reference frames that are used in computational vision for object detection. They are the smallest possible box that can fit around the target. Bounding boxes completely cover the target and simplify the classification of the object of interest. They are used primarily in object detection to locate and identify the position and type of multiple objects in an image [10]. An axis-aligned bounding box, or AABB, has axes aligned with the coordinate axes. In 2D, an AABB is a rectangle with faces parallel to the x and y axes. The main characteristics of AABBs are [23]:

- Orientation: An AABB is defined strictly by its position and size, without any rotation component. AABB makes it suitable for objects aligned with the coordinate axes, simplifying calculations and implementations in many computer vision tasks.
- Specification: AABBs are typically specified by two parameters: the minimum and maximum coordinates, which define the corners of the rectangle along the x- and y-axes. AABBs can be represented as (xmin, ymin) and (xmax, ymax).
- Applications: They are particularly effective in scenarios where objects align with the axes of the image, such as scanned documents or images where the layout is standardised and grid-aligned.
- Benefits: Using AABBs simplifies the computation involved in object detection, as it avoids the complexities of rotational alignment. This simplification can lead to faster processing times and reduced computational overhead in many applications.
- Challenges: While more straightforward to process, AABBs may not encapsulate the shape and orientation of objects, mainly when objects are not axis-aligned. This lack could lead to less efficient use of space within the bounding box and result in less accurate object detection in specific contexts.

An orientated bounding box is another possible representation where the box edges on different sides are still perpendicular, but not necessarily coordinate system aligned. The main characteristics of the OBBs are [13,21]:

- Orientation: An OBB is defined by the position, width, height, and rotation angle. OBB allows the box to fit the object's contour, especially for objects with non-vertical or horizontal orientations.
- Specification: An OBB is commonly specified by five parameters: the centre coordinates (x, y), the width, height, and the rotation angle. Alternatively, it can be defined by the four vertices of the rectangle.
- Applications: They are instrumental in images where objects are oriented arbitrarily, such as satellite images, aerial photographs, or scenarios where objects are not aligned with the image axes.
- Benefits: Using OBBs can lead to greater accuracy in object detection, as they allow for a more exact match of the object's shape and orientation.
- Challenges: Processing OBBs is generally more complex than processing AABBs, requiring rotation consideration. This complexity can include more complicated geometric transformations and adjustments to how deep learning models process this information.

Polygon annotation involves outlining objects in images using polygonal shapes, enabling algorithms to differentiate between objects and backgrounds. It is usually used for image segmentation, which seeks to find the pixel boundaries between objects, whereas object detection aims to locate objects. The closest-to-ground truth polygons often require an annotator to identify many points around a target of interest to fit its shape. This requirement makes polygons the most time-consuming, and thus the most expensive of the bounding-box annotations. They are more time-efficient to produce than polygons, requiring only two clicks from the annotator. However, they may encapsulate many pixels outside of the target area [3]. The main characteristics of polygonal annotations are [9]:

- Orientation: Polygonal annotations are defined by vertices that outline an object's exact shape and are not limited to rectangular forms. This flexibility allows for a more accurate representation of complex and irregular object contours, crucial in diverse imaging contexts.
- Specification: Polygonal annotations are specified by a list of vertex coordinates that define the polygon enclosing the object. The number of vertices can vary depending on the complexity of the object's shape.
- Applications: These annotations are especially useful in images with irregularly shaped objects, such as those found in natural environments, medical imaging, and artistic fields, where precise object boundaries are necessary.
- Benefits: Using polygonal annotations leads to higher object delineation and segmentation precision. This precision is beneficial for applications requiring detailed analysis and processing of object shapes, contributing to more accurate object detection and classification.
- Challenges: The main challenge with polygonal annotations is the increased complexity in both annotation and processing. Creating and handling polygonal shapes requires more sophisticated tools and algorithms, potentially

increasing the computational load compared to more straightforward bounding box methods.

3 Proposed Methodology

This section presents a practical and highly efficient methodology that converts axis-aligned bounding box annotations to polygonal and oriented bounding box annotations. This innovative process empowers computer vision researchers and application developers, offering significant implications for a wide range of applications.

Our conversion methodology uses the flexible and user-friendly Segment Anything Model (SAM) [19], an advanced deep learning architecture designed for image segmentation. SAM combines transformer-based models and convolutional neural networks (CNNs) to achieve high accuracy in various contexts and applications. The key components of SAM are:

– Backbone Network: This is typically a deep CNN (e.g. ResNet) or a transformer model (e.g., Vision Transformer), responsible for extracting high-level features from input images and capturing long-range dependencies and contextual information.
– Multi-Scale Feature Extraction: Using feature pyramid networks (FPN) or similar techniques to process features at various resolutions, ensuring accurate segmentation of objects of different sizes.
– Transformer-Based Encoder-Decoder: The encoder processes multiscale features into a compact representation, and the decoder uses this to predict segmentation masks, leveraging the self-attention mechanism of transformers to effectively model relationships within the image.
– Mask Prediction Head: Generates final segmentation masks, often including upsampling layers to produce high-resolution masks, refining segmentation boundaries for high precision.

SAM is trained using a combination of loss functions, such as cross-entropy loss and dice loss, tailored for segmentation tasks. The training involves large-scale annotated datasets to ensure the model generalises well to different segmentation challenges. Figure 1 shows the effectiveness of SAM with an aerial image of a transmission tower, where the original image is on the left and the segmented image, produced using SAM, is on the right.

The central idea is to use SAM to create a segmentation mask, then find the polygonal annotation through the vertices of the polygon formatted by the corners of this mask, and then the minimal rotated rectangle that contains all the vertices of the polygon to find the orientated bounding box annotation. The Fig. 2 shows the flow to the training of models, including the methodology steps for converting AABB to polygonal and OBB annotations. There are four steps in this methodology: (1) Generate Polygon;(2) Create Polygonal Annotations; (3) Generate Rotated Rectangle; (3) Create OBB Annotations.

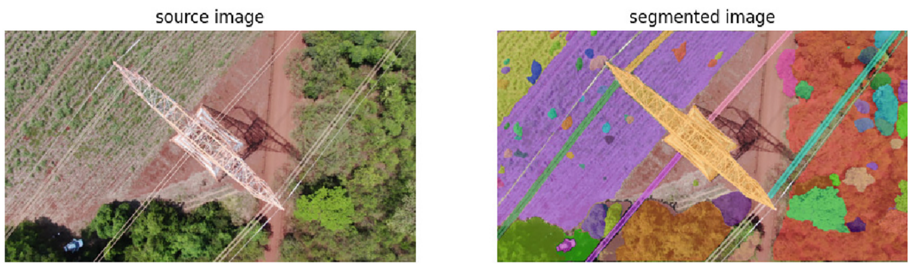

Fig. 1. Comparison Original Segmented Image.

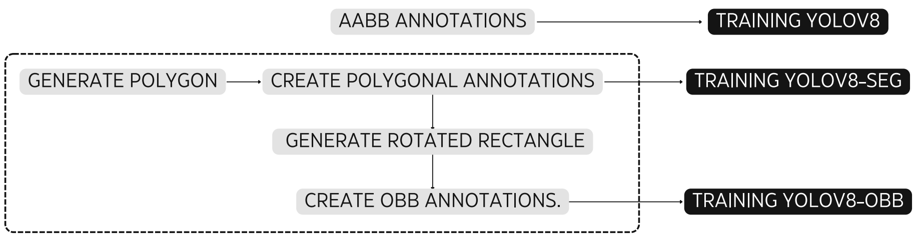

Fig. 2. Conversion to Annotations Methodology.

Generate Polygon. This first step has three activities: (i) Convert the annotations format to the format required by SAM. Usually, AABB coordinates are in the format $(x_{center}, y_{center}, w, h)$, where x_{center} and y_{center} are the coordinates of the central point of the bounding box, and w and h are the weight and height of the bounding box, respectively. As output, we have the coordinates in the format required by SAM, that is $(x_{min}, y_{min}, x_{max}, y_{max}$, where x_{min} and y_{min} are the coordinates of the minimal vertex of the bounding box, and x_{max} and y_{max}, the coordinates of the maximum vertex; (ii) Apply SAM to generate polygons. The SamPredictor, an SAM class, provides an interface to the model to prompt the model. This interface allowed us to set an image using the `set_image` method, which calculates the necessary image embeddings. We provide the AABB annotations as prompts via the predict method to predict masks from those prompts efficiently. As output, we have the polygon coordinates;(iii)Visualise results, drawing the polygons over objects to check whether the objects are entirely inside of polygons.

Create File Polygon Annotation. In this step, we access the AABB annotations folder. Then, we open each text file annotation one by one and read the file annotation line by line in order to convert it to a polygon annotation. Most object and segment detectors use a format that includes one text file per image, which contains the annotations and a numeric label representation. The annotations are standardised to range from 0 to 1, making them easy to manipulate even after image resizing or distortion. As a result, we receive the polygon annotations folder as output.

Generate Rotated Rectangle. In this step, we use the polygon and the dimensions of the image to find the minimal rotational rectangle that contains this polygon. OpenCV [8] provides a function `cv2.minAreaRect()` for determining the minimum area rotated rectangle. This function requires a 2D point set as input and returns a `Box2D` structure containing the following attributes: (x_{center}, y_{center}, w, h, angle of rotation). However, to draw a rectangle, we require the coordinates of its four corners. Therefore, to convert the `Box2D` structure to four corner points, OpenCV offers another function, `cv2.boxPoints()`. This function takes the `Box2D` structure as input and returns the coordinates of the four corner points. These points are ordered clockwise, starting from the point with the highest y coordinate. Before drawing the rectangle, it is necessary to convert the coordinates of the four corners to integers.

Once the coordinates of the four corners have been obtained, the rectangle can be quickly drawn. As the last activity of this step, we visualise results, drawing the rotated rectangles over objects to check whether the objects are entirely inside of rectangles.

Create File OBB Annotation. In this step, we follow a process similar to the second step. We access the AABB annotations folder and open each text file annotation one by one. Then, we read the file annotation line by line and convert it to a nan OBB annotation. The output we achieve from this process is the OBB annotations folder.

3.1 Comparison with Existing Techniques

Recent advances in object detection, such as Faster R-CNN, Mask R-CNN, and RetinaNet, have shown high performance in detecting objects aligned with image axes but struggle with arbitrary orientations due to their reliance on AABBs [16,20,24]. To address this, methods such as rotation-invariant CNNs [11] and orientated R-CNN [29] have been developed, which handle OBBs in aerial imagery more effectively. However, these methods still require substantial manual annotation and model retraining.

Our approach differentiates itself by using the Segment Anything Model (SAM) [19] and YOLOv8 [18] to automate the conversion of AABBs into polygonal and OBBs, improving annotation precision without modifying the model architecture. This reduces manual effort and maintains high performance in varied contexts, unlike models that require extensive tuning for different datasets [11,21].

4 Experimental Study

This section presents an application of our methodology for converting annotations in a real-world dataset of aerial images of transmission towers from Transmissora Aliança de Energia Elétrica S.A. (TAESA), Brazil. Our goal in this experimental study is to validate the converted annotations by testing them on the most advanced YOLO models. These models were trained within the Ultralytics YOLOv8.0.42 environment, using the power of Python version 3.10.12 and PyTorch version 2.0.1. This setup was specifically implemented on NVIDIA A100-SXM4-40GB graphics processing units, each equipped with a substantial 40,514 MiB of memory. The inclusion of support for CUDA 12.1 further enhanced the efficiency and speed of our training process.

There are 1,144 files of AABB annotations of the aerial images in the TAESA dataset, splitting into 914 for training, 114 for validation, and 115 for testing. This extensive data set encompasses a variety of images obtained from various transmission sites across a Brazilian municipality, each characterised by different soil compositions and vegetation arrangements. The objective is to monitor the maintenance status of the towers to avoid accidents. Numerous practitioners meticulously annotated the images, outlining the presence of transmission towers (the focal objects of interest), to mitigate the possibility of oversight or inconsistency in the annotations, thus increasing the credibility of the data set for future analytical efforts [7].

Figure 3 provides a visual representation of the application of the methodology, showing an example of an image from the TAESA dataset. In the first line and column, a picture shows the original image with the AABB overlaid. In the first line, in the second column, a picture shows the image with the segmented mask. In the second line, in the first column, a picture shows the polygon found from the contours of the image. In the second line and column, a picture shows the original image with the OBB overlaid.

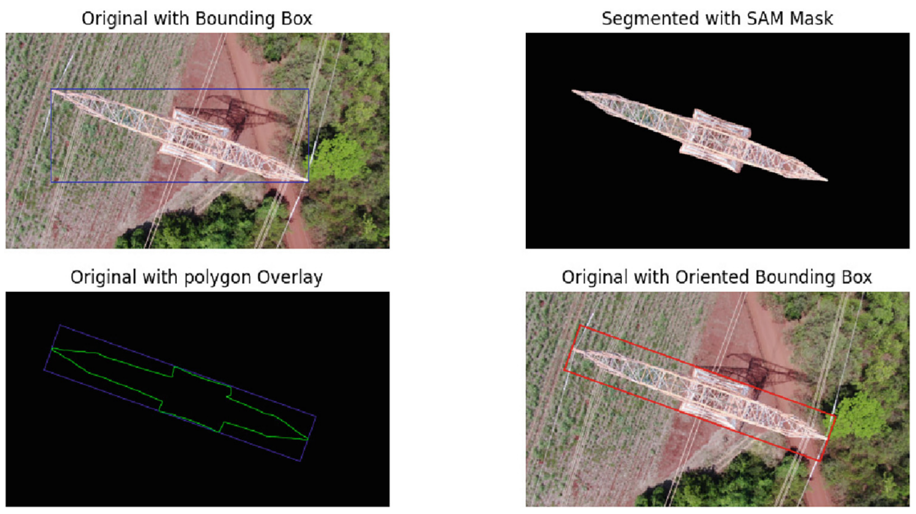
Original with Bounding Box
Segmented with SAM Mask
Original with polygon Overlay
Original with Oriented Bounding Box

Fig. 3. Example of the Application of the Methodology.

5 Validation

We train models with YOLOv8-seg and YOLOv8-ob to validate the polygonal and OBB annotations generated by our conversion methodology. We employed the most streamlined variants of these models, namely YOLOv8n-seg and YOLOv8n-obb, utilising their standard configurations without modifications. These models were trained within the Ultralytics YOLOv8.0.42 environment, using Python version 3.10.12 and PyTorch version 2.0.1, which includes support for CUDA 12.1. This setup was implemented on NVIDIA A100-SXM4-40GB graphics processing units equipped with 40,514 MiB of memory.

We train models with YOLOv8, YOLOv8-seg, and YOLOv8-obb to validate the polygonal and OBB annotations generated by our conversion methodology. We employed the most streamlined variants of these models, namely YOLOv8n, YOLOv8n-seg, and YOLOv8n-obb, utilising their standard configurations without modifications. Figure 4 shows a sample of images with their respective bounding boxes and masks with the three YOLO models, where it is possible to verify that in all models, the bounding boxes and masks correctly overlap the transmission towers.

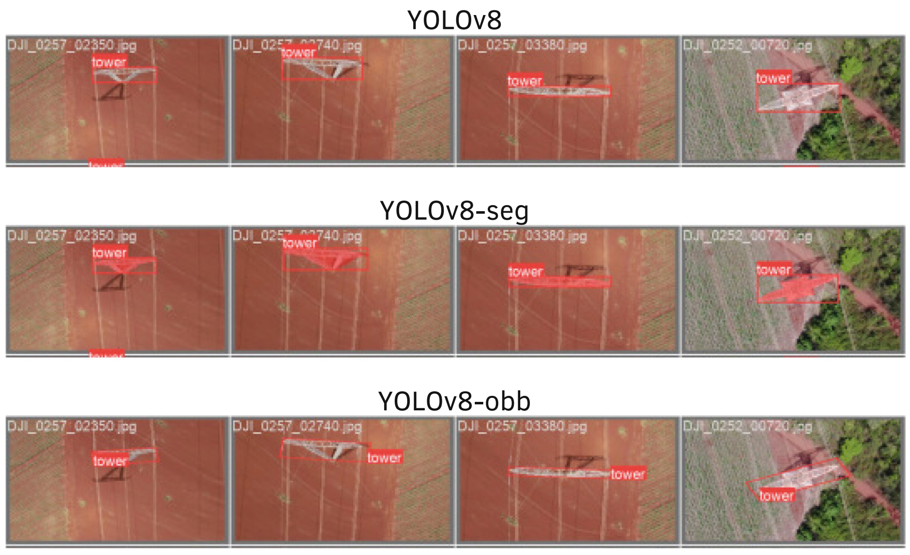

Fig. 4. Detailed Visualisation of Bounding Boxes and Masks on Objects.

Table 1 presents performance metrics for three configurations of the YOLO object detection models-YOLOv8, YOLOv8-seg, and YOLOv8-obb. The evaluated metrics include Precision, Recall, mAP50, and mAP50-95, which are crucial for assessing the efficacy of bounding box and polygon annotations under different model configurations.

The YOLOv8 model exhibits the highest precision (1.000) and a high recall (0.991), indicating that almost all predictions made by the model are correct and that the majority of the objects of interest were detected. This performance is superior compared to YOLOv8-seg and YOLOv8-obb, which show significantly lower precision and recall values. This can be attributed to the simplicity of the YOLOv8 model in handling regular bounding boxes, whereas the other models are specialised in segmentation and oriented bounding boxes, which are more complex tasks.

The mAP50 of YOLOv8 (0.995) is extremely high, suggesting an almost perfect agreement between the predicted and actual annotations at a 50% IoU threshold. In contrast, the mAP50 values for YOLOv8-seg and YOLOv8-obb are lower (0.881 and 0.719, respectively), highlighting the additional challenges associated with segmentation and oriented bounding box tasks.

The mAP50-95, which evaluates the accuracy of annotations across a range of IoU thresholds from 50% to 95%, shows a significant decrease in values, especially for YOLOv8-obb (0.347). This result indicates that the performance of specialised models tends to drop at higher IoU thresholds, which may reflect the complexity and variability of object shapes in the annotated data.

Compared with other studies in the literature, it is evident that YOLOv8 offers robust performance in terms of precision and recall, making it well suited for object detection tasks with regular bounding boxes. However, for applications that require precise segmentation or orientated bounding box detection, improvements in the YOLOv8-seg and YOLOv8-obb models are necessary. This variation in performance underscores the importance of selecting the appropriate model based on the type of annotation and the application context. Moreover, the high precision and recall of YOLOv8 are promising for applications in infrastructure monitoring and maintenance, such as in the case of TAESA's transmission towers, where accurate detection can lead to significant reductions in maintenance costs and accident prevention.

The limitations of our approach include dependency on high-quality initial annotations and potential inefficiencies in handling highly irregular object shapes or low-contrast objects. Future work should focus on integrating additional contextual information and exploring advanced augmentation techniques to enhance model robustness. The results obtained pave the way for future research focused on improving segmentation and oriented bounding box models, with an emphasis on data augmentation, fine-tuning models, and incorporating advanced deep learning methodologies to potentially enhance the performance of YOLOv8-seg and YOLOv8-obb models.

Table 1. Performance metrics of various YOLO model configurations

Model	Precision	Recall	mAP50	mAP50-95
YOLOv8	1.000	0.991	0.995	0.969
YOLOv8-seg	0.883	0.868	0.881	0.799
YOLOv8-obb	0.747	0.712	0.719	0.347

6 Conclusion

Our automated methodology for converting AABBs to polygonal and OBBs has broader applications beyond aerial imagery for electrical transmission towers. It can be adapted to fields such as urban planning, where the precise detection of building footprints from satellite images is crucial for infrastructure development [22]. In disaster response, rapid annotation of affected areas from aerial or satellite imagery can accelerate rescue operations [15]. In addition, environmental monitoring, including wildlife tracking and vegetation analysis, benefits from scalable methods to manage large data sets [25]. By improving data annotation quality, our approach improves the effectiveness of machine learning models in various applications.

Acknowledgment. The authors would like to express their sincere gratitude to Transmissora Aliança de Energia Elétrica S.A., Brazil, for their valuable support in conducting this research (PD-07130-0059/2020 and PD-07130-6059/2022), with resources from the R&D Program of the Brazilian Electricity Regulatory Agency (ANEEL). The support provided by all individuals involved in this collaboration, with their insightful discussions, guidance, and expertise, contributed to the completion of this study.

References

1. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation (2015)
2. Xia, G.S., et al.: DOTA: a large-scale dataset for object detection in aerial images (2018)
3. Mullen Jr, J.F., Tanner, F.R., Sallee, P.: A Comparing the effects of annotation type on machine learning detection performance (2019)
4. Tian, Z., Shen, C., Chen, H., He, T.: FCOS: Fully Convolutional One-Stage Object Detection (2019)
5. Kirillov, A., Girshick, R., He, K., Dollar, P.: Panoptic feature pyramid networks (2019)
6. Ma, J., Ushiku, Y., Sagara, M.: The effect of improving annotation quality on object detection datasets: a preliminary study (2022)

7. Menezes, A.G., Peterlevitz, A.J., Chinelatto, M.A., de Carvalho, A.C.: Efficient Parameter Mining and Freezing for Continual Object Detection. SCITEPRESS (2024)
8. Bradski, G.: The opencv library. Dr. Dobb's J. Softw. Tools Prof. Program. **25**(11), 120–123 (2000)
9. Buslaev, A., Iglovikov, V.I., Khvedchenya, E., Parinov, A., Druzhinin, M., Kalinin, A.A.: Albumentations: fast and flexible image augmentations. Information **11**(2), 125 (2020)
10. Caldwell, D.R.: Unlocking the mysteries of the bounding box. Coordinates: Online J. Map Geogr. Round Table Am. Libr. Assoc. Series A, 1–20 (2005)
11. Cheng, G., Zhou, P., Han, J.: Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images. IEEE Trans. Geosci. Remote Sens. **54**(12), 7405–7415 (2016)
12. Ding, J., et al.: Object detection in aerial images: a large-scale benchmark and challenges. IEEE Trans. Pattern Anal. Mach. Intell. **44**(11), 7778–7796 (2021)
13. Fang, Z., Ren, J., Sun, H., Marshall, S., Han, J., Zhao, H.: SAFDet: a semi-anchor-free detector for effective detection of oriented objects in aerial images. Remote Sens. **12**(19), 3225 (2020)
14. Gupta, A., Dollár, P., Girshick, R.B.: Lvis: a dataset for large vocabulary instance segmentation. In: Conference on Computer Vision and Pattern Recognition, pp. 5351–5359 (2019)
15. Gupta, R., et al.: Creating xbd: a dataset for assessing building damage from satellite imagery. In: Conference on Computer Vision and Pattern Recognition Workshop, pp. 10–17 (2019)
16. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. IEEE Trans. Pattern Anal. Mach. Intell. **42**(2), 386–397 (2017)
17. Jain, A., et al.: Ai-enabled object detection in UAVs: challenges, design choices, and research directions. IEEE Netw. **35**(4), 129–135 (2021)
18. Jocher, G., Chaurasia, A., Qiu, J., et al.: YOLOv8: the latest version of YOLO models. Ultralytics (2023)
19. Kirillov, A., et al.: Segment anything (2023)
20. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. Int. Conference on Computer Vision, pp. 2980–2988 (2017)
21. Liu, Y., Jiang, W.: Oii: an orientation information integrating network for oriented object detection in remote sensing images. Remote Sens. **16**(5), 731 (2024)
22. Maggiori, E., Tarabalka, Y., Charpiat, G., Alliez, P.: Convolutional neural networks for large-scale remote-sensing image classification. IEEE Trans. Geosci. Remote Sens. **55**(2), 645–657 (2017)
23. Mojtahedi, M.: A safety warning algorithm based on axis aligned bounding box method to prevent onsite accidents of mobile construction machineries. Sensors **21**(21), 7075 (2021)
24. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. 91–99 (2015)
25. Russwurm, M., Korner, M.: Convolutional lstms for cloud-robust segmentation of remote sensing imagery. arXiv preprint arXiv:1807.00347 (2018)
26. Saini, M.K., Goel, N., Shekhawat, H.S., Mauri, J.L., Singh, D. (eds.): Fast Rotated Bounding Box Annotations for Object Detection. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-43605-5_8
27. Singh, P.: Systematic review of data-centric approaches in artificial intelligence and machine learning. Data Sci. Manag. **6**(3), 144–157 (2023)

28. Lin, T.-Y., et al.: Microsoft COCO: common objects in context. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8693, pp. 740–755. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10602-1_48
29. Xia, X., Liu, Y., Sun, Z.: Oriented r-cnn for object detection. IEEE Trans. Pattern Anal. Mach. Intell. **43**(10), 3697–3705 (2021)
30. Zha, Det al.: Data-centric artificial intelligence: a survey. arXiv preprint arXiv:2303.10158 (2023)