scientific reports

OPEN

Inspection of railway catenary systems using machine learning with domain knowledge integration

Kacper Marciniak[1]✉, Paweł Majewski[2] & Jacek Reiner[1]

Railway catenary system inspection is a critical task where high accuracy and reliability are essential to ensure operational efficiency and safety. This objective is achieved by assessing the technical condition of the infrastructure and maintaining a comprehensive inventory of its components. The application of machine learning methods to this problem is non-trivial, due to various constraints, including the cost of data acquisition. This paper presents innovative solutions leveraging domain knowledge to significantly improve the inference quality of machine learning models using existing training data. Key innovations include a two-stage approach and clustering of selected objects to extract regions of interest (ROI), dynamic confidence score weighting, and ROI masking, aimed at reducing false positives and enhancing precision. Additionally, the system was extended with ensemble learning methods and custom test-time augmentations (TTA). Proposed methods substantially improve metrics such as AP50, precision, recall, and F1-score, particularly in detecting small and hard-to-spot catenary components such as insulators. Notably, the proposed enhancements were optimized to mitigate processing time increases, enabling their application in industrial settings. The results demonstrate the effectiveness of integrating domain knowledge into the process of machine vision inspection, achieving an improvement in the F1-score metric from 61.97 (0.59) to 82.53 (0.38) compared to the baseline single-model approach while maintaining practical runtime constraints for real-world overhead catenary system inspection.

Keywords Machine vision, Machine learning, Vision inspection, Railway, Domain knowledge

Railways are a key component of modern transportation systems, particularly in the European Union, where in 2023 passenger transport reached 429 billion passenger-kilometers, and freight transport accounted for 378 billion tonne-kilometers[1]. As one of the most environmentally friendly modes of transport, railways significantly contribute to reducing CO_2 emissions compared to road transport, aligning with EU sustainability goals. Beyond their environmental benefits, railways represent a critical element of national and international infrastructure, ensuring economic stability, territorial connectivity, and resilience in times of crisis. They play a strategic role in national security policies, enabling the rapid and efficient mobilization of troops, equipment, and essential supplies during emergencies or conflicts[2]. Therefore, ensuring reliability, security, and resilience of railway infrastructure is not only an economic necessity but also a strategic priority for governments and infrastructure operators.

The railway networks are extensive—Poland alone has 18,600 km of actively operated railway lines managed by Polish Railway Lines (PKP PLK)[3]. The railway infrastructure, essential for the safe transport of passengers and goods, is highly complex and undergoes gradual wear and tear during operation. Maintaining it in good condition involves inspection, repairs and modernization, which is crucial for passengers, goods, and environmental safety. Additionally, changing climate conditions, such as extreme temperatures or heavy rainfall, accelerate the degradation process, further emphasizing the need for frequent and reliable inspections. To manage such a vast and intricate network, Geographic Information Systems (GIS) are used to digitize infrastructure assets. Updating these digital maps demands efficient methods for data acquisition and processing, which serves as the motivation for this work. This article focuses on the automation of railway infrastructure inspection, with a particular emphasis on the assessment of overhead line components.

[1]Faculty of Mechanical Engineering, Wrocław University of Science and Technology, ul. Łukasiewicza 5, 50-371 Wrocław, Poland. [2]Faculty of Information and Communication Technology, Wrocław University of Science and Technology, ul. Janiszewskiego 11/17, 50-372 Wrocław, Poland. ✉email: kacper.marciniak@pwr.edu.pl

Elements of the railway overhead caternary system are particularly critical components of railway infrastructure. They enable the continuous and safe electrical contact between the train's pantograph and the overhead power supply—failures in these elements can lead to power outages, mechanical damage, or even derailments at high speeds. Consequently, there is a strong need for precise and reliable methods for monitoring and assessing their condition.

Given their role and specific characteristics, inspecting elements of the overhead catenary system is not a trivial task. The main challenges in the inspection of power supply infrastructure include the small size of components—such as insulators, switches, and disconnectors—which significantly complicates their detection, especially when captured from aerial imagery where they may blend with the background of the railway tracks. Moreover, their placement several meters above track level requires careful configuration of the acquisition system. Finally, the presence of high-voltage power introduces additional safety considerations, necessitating the maintenance of a safe distance between the imaging equipment and the overhead wires.

Currently, most inspection tasks are performed visually. Trained personnel walk along selected track sections and assess the technical condition of components according to established procedures. This approach is characterized by low efficiency and high time consumption. Therefore, some efforts have been made to develop machine vision systems for the automatic (or partially automatic) inspection of railway infrastructure. These efforts can be categorized on the basis of two factors: the approach to data acquisition and the data processing methods used.

In terms of acquisition methods, two main groups can be distinguished: image acquisition from traction vehicles or UAVs (unmanned aerial vehicles). Traction vehicles are widely used in the inspection of railway infrastructure. Recently, these vehicles have been equipped with vision systems to perform tasks such as detecting rail fasteners[4], evaluating the technical condition of cantilevers and insulators on rail overhead lines[5], and assessing the condition of tunnel walls on high-speed rail lines[6]. Using vehicles as platforms for vision systems offers several advantages such as: high payload capacity (allowing for multiple cameras/sensors and other necessary equipment) and ability to position cameras close to the elements under inspection (rails, sleepers, overhead line components). However, there are notable disadvantages, such as high operating costs, the requirement for highly specialized personnel (train drivers and vehicle operators) and track occupancy during inspections. These challenges can be mitigated by using UAVs as platforms for the inspection systems developed.

UAV platforms equipped with various sensors have been widely adopted for telemetric and scanning tasks, including railway infrastructure imaging. Infrastructure operators such as Deutsche Bahn (Germany), Network Rail (United Kingdom), and CAF Signalling (Spain) have successfully deployed UAVs for monitoring infrastructure conditions[7]. Studies confirm the feasibility of UAV-based vision systems for railway inspection within existing regulations[8]. Recent advancements include drone-based rail surface defect inspections[9], sleeper condition assessment[10], rail positioning measurements using photogrammetry[11], and vegetation and obstacle detection along railway lines[12].

Inspection systems relying solely on classical image processing methods are not commonly used for complex railway infrastructure due to their limited flexibility. Classical computer vision techniques are primarily applied to well-defined tasks, such as measuring rail-to-sleeper fasteners[13]. A more effective approach involves hybrid solutions that integrate classical machine vision (MV) methods with machine learning (ML). For instance, a system for rail position detection[14] utilizes classical methods such as binarization, segmentation, and edge detection to identify object positions and extract features (e.g., height, length, eccentric color). These features are then classified using a k-Nearest Neighbors (k-NN) algorithm, significantly improving precision.

Another example is a system designed for inspecting insulators in railway overhead lines for high-speed rail networks[15]. The authors proposed a two-stage detection process: (1) identifying regions of interest (ROIs) containing individual rods of the cantilever structure through image binarization and segmentation, and (2) localizing insulators within these ROIs using discriminatively trained parts-based models[16].

Although these solutions achieved high effectiveness under controlled conditions, they also exhibit several limitations. A major drawback is their low flexibility, which restricts their applicability. For example, the described insulator inspection system requires nighttime data acquisition with additional illuminators to enhance object visibility and contrast. Any variations in acquisition conditions or differences in analyzed objects (e.g., different insulator or cantilever types) may lead to a significant decline in inference quality. Consequently, such systems are limited to a narrow range of railway infrastructure and specific data acquisition settings.

Another category of approaches that has found widespread application in quality inspection and infrastructure inventory management is based on deep learning (DL). Within this category, off-the-shelf models such as Faster R-CNN[12,17,18] and YOLO (You Only Look Once)[19–21] dominate, frequently used for detecting infrastructure elements. Notable custom solutions also exist, such as the Real-Time Rail Recognition Network (TriRNet)[22], a deep architecture designed for detecting the positions of railway tracks from UAV-acquired images.

Deep learning methods are widely regarded as state-of-the-art solutions for the inspection of railway infrastructure. They offer greater flexibility and robustness to data variability compared to the previously described approaches, enabling automated detection and analysis of complex infrastructure components with higher accuracy—even under challenging environmental conditions that often hinder traditional rule-based or manual approaches. By learning patterns directly from data, DL systems are capable of detecting subtle defects and generalizing across various infrastructure designs. This capability translates into faster, safer, and more consistent inspections at scale, significantly improving the efficiency and reliability of maintenance workflows. Moreover, maintaining and monitoring a DL-based system after deployment is generally easier and more standardized. Unlike classical methods, which often require extensive manual adjustments or rule redesign, DL models can be updated or fine-tuned with minimal intervention, making them more scalable and adaptable to changing operational conditions[23]. However, DL methods are not without limitations. One of the most significant challenges for supervised deep learning models is the requirement for sufficiently large and diverse labeled training datasets. The quality of the available data directly affects the performance and generalization capabilities of the developed models. However, the processes of data acquisition, proper dataset preparation, and model training are both costly and time-consuming.

As a result, there is a growing demand for methods that enhance inference quality without necessitating the creation of new, larger datasets. One effective approach to improving the performance and generalization capabilities of deep learning models is the incorporation of domain knowledge—that is, expert knowledge specific to a given problem or application domain[24]. Several strategies exist for integrating domain expertise into deep learning pipelines. These include, for instance, embedding domain knowledge during the architecture design stage, such as implicitly reflecting temporal dynamics[25], leveraging prior knowledge from established diagnostic systems to enhance novel systems and enable cross-modality knowledge transfer[26], applying conditional alignment strategies[27], or incorporating known characteristics during data preprocessing and structured feature learning using Transformer-based models[28].

Considering the aforementioned challenges related to data acquisition, as well as the additional need to improve the interpretability of automated inspection systems—for instance, by decomposing them into smaller, more comprehensible modules—the authors propose a multi-stage processing framework that explicitly leverages domain-specific knowledge in the context of railway infrastructure inspection. This framework includes: (1) filtering detections based on their spatial distance from the tracks; (2) clustering detections to form "traction groups" and define regions of interest (ROIs) for downstream processing; (3) masking ROIs using outputs from previous detection stages to limit irrelevant context; (4) employing a custom test-time augmentation (TTA) strategy tailored to the structural characteristics of the problem. The proposed enhancements are designed to significantly boost the inference quality of deep learning models by providing an additional layer of contextual filtering and optimization. Importantly, these techniques are model-agnostic and can be implemented independently of the core architecture, ensuring seamless integration into a wide range of existing railway inspection pipelines.

Materials and methods

Problem definition

The research problem addressed in this article focuses on developing methods to efficiently and accurately detect key elements of railway infrastructure, with particular emphasis on overhead line components, using images obtained from unmanned aerial vehicles (UAVs). The need for such methods arises from the requirement to reliably identify and monitor the condition of various infrastructure components, ensuring the safe and efficient operation of the railway system. The elements selected for this purpose—such as overhead lines, poles, and associated structural components—are listed in Table 1.

The methods proposed in this paper form the foundation of a larger system aimed at automating the inventorying and inspection of railway infrastructure maintenance. By employing rapidly deployable UAVs, the system can acquire high-quality images, even in hard-to-reach or challenging areas, thereby reducing the time, cost, and risks typically associated with traditional inspections. This approach not only improves the efficiency of infrastructure monitoring but also facilitates real-time decision-making by providing accurate data on the condition of the railway system.

The proposed approach utilizes UAVs exclusively for image acquisition. All processing is performed offline on a workstation after the flight images have been captured. This method enables image processing at full resolution, avoiding the hardware limitations imposed by restricted computing power and power supply constraints. The inference results and metadata are integrated with the commercial GIS-based DRONonLine platform (*GISonLine, Poland*), which visualizes the UAV flight path, the locations of the captured images, as well as information regarding the location and condition of railway catenary system components. The overall scheme of the proposed solution is shown in Fig. 1.

Input data

Data acquisition was performed using an UAV equipped with a ZENMUSE P1 camera (*SZ DJI Technology Co. Ltd., China*), capable of capturing images at a resolution of 8192×5460. The camera was positioned at an angle of approximately 45 degrees to the terrain. Due to regulations governing railway operations, UAV flights were conducted at a minimum altitude of 30 m.

The prepared dataset comprises 733 high-resolution images of railway infrastructure, captured during seven flights over various railway lines and stations (Fig. 2):

1. Nowy Targ—railway station, summer, sunny;

Infrastructure	Description	Class names
Support structures	Individual and group structures such as frames and poles	Pole, horizontal bar
Registration arms/cantilevers	Load-bearing structure attaching the suspension line to the substructure	Mount
Insulators	Insulators used in cantilevers and in the mounting of section power cables	Insulator
Power switches and disconnectors	Equipment to disconnect the power supply to a section of the overhead line	Electrical component
Railroad switches	–	Railroad switch
Semaphore	Colour light signal	Semaphore

Table 1. Relevant groups of railway infrastructure facilities.

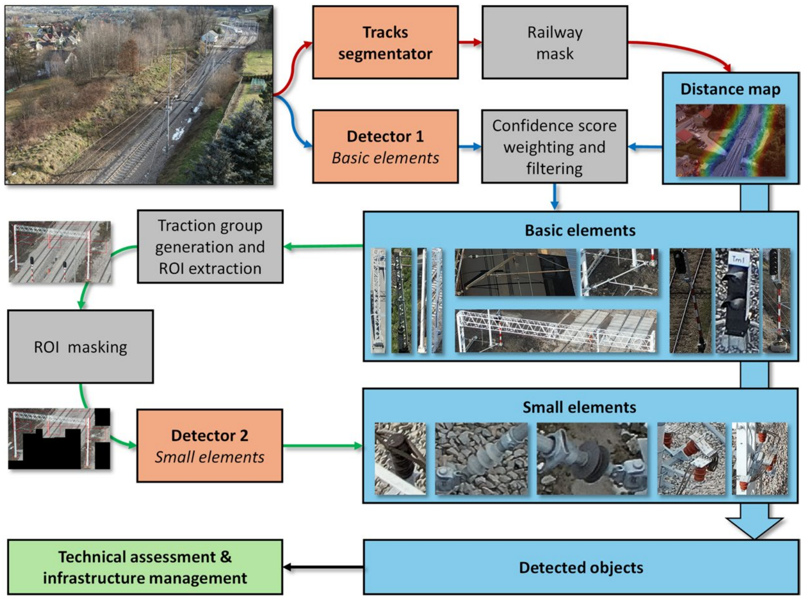

Figure 1. General scheme of the proposed image processing system. *(original images courtesy of GISonLine).*

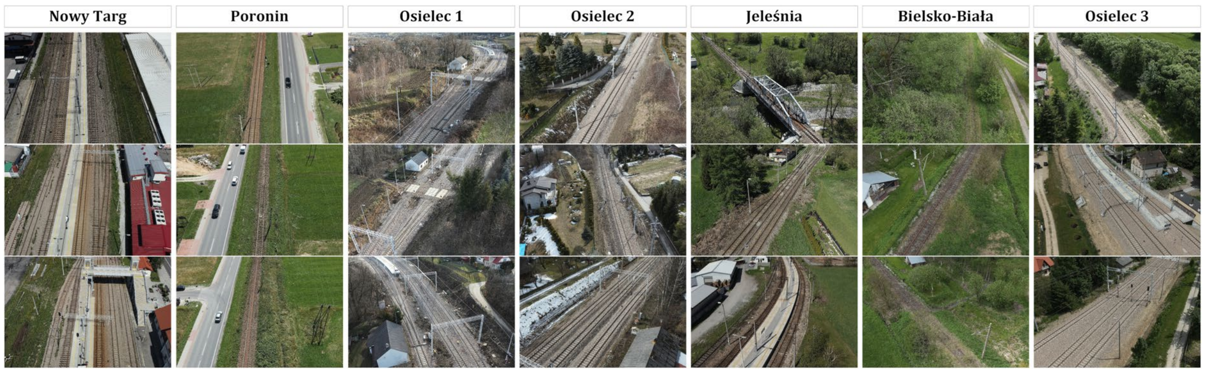

Figure 2. Review of images taken during UAV flights (original images courtesy of GISonLine).

2. Poronin—railway line, summer, sunny;
3. Osielec 1—station and railway line, winter, cloudy;
4. Osielec 2—station and railway line, winter, sunny;
5. Jeleśnia—station and railway line, spring, sunny;
6. Bielsko-Biała—abandoned railway line, spring, cloudy;
7. Osielec 3—station and railway line, summer, sunny.

In total, 15,335 objects were labeled in the images. The dataset was divided into training and validation subsets in an 80/20 ratio.

As shown in Fig. 3, the labeled objects can be categorized into two groups based on the size of their bounding boxes: large objects (*horizontal bar*, *pole*, *mount*, *railway switch*, *semaphore*) and small objects that are challenging to detect in the images (*insulator*, *electrical component*). Three datasets were created: (1) containing only large objects (*'basic elements'*), (2) containing only small objects (*'small elements'*), and (3) containing both groups of objects. Due to the dimensional characteristics of the images and objects, input image scaling and tiling were applied during the dataset creation process (Table 2).

Two stage processing

The most straightforward approach to object detection would be to employ a single model trained on all classes described in "Problem definition". However, due to differences in the size and characteristics of the objects (Fig. 3), a multistage approach was adopted, utilizing two models. The first model detects objects belonging to the "basic elements" group, while the second focuses on "small elements". Each model employs its own preprocessing parameters, identical to those used during the creation of the training dataset (Table 2). These parameters are as follows:

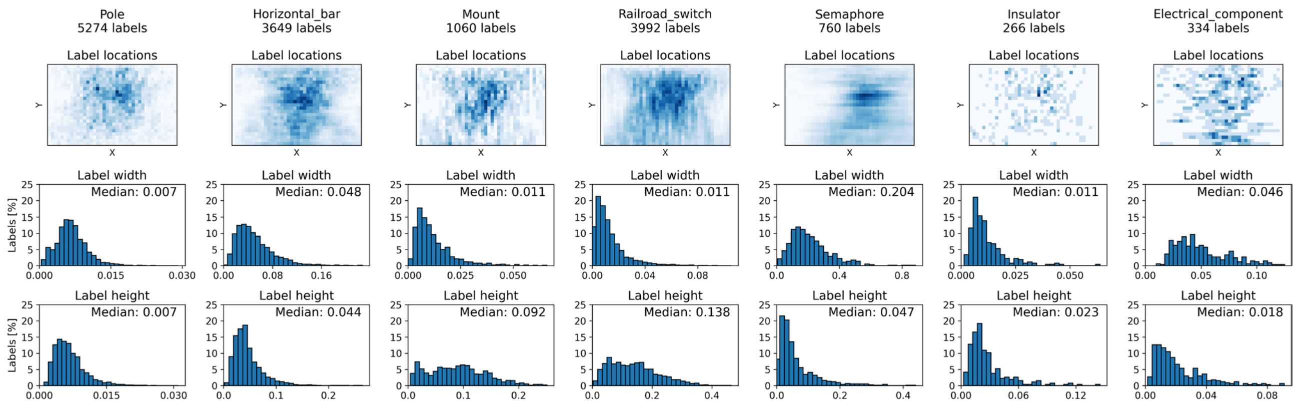

Figure 3. Analysis of the size and location of labels.

Group name	Classes	Tile size	Overlap	Scale
Basic elements	All except *insulator* and *electrical component*	1216	35%	0.35
Small elements	*Insulator, electrical component*	1216	25%	1.0
All elements	All classes	1216	35%	0.35

Table 2. Object groups and their scaling and tiling parameters for input images used during data set creation.

- Tile size—the dimensions of square tiles used for image tiling.
- Overlap—the degree of overlap between adjacent tiles.
- Scale factor—the scaling factor applied to resize the image before tiling.

The adoption of two-stage processing significantly increases processing time, as images are tiled twice and transferred to the corresponding models. Furthermore, similar to the one-step approach, there is a considerable risk of false-positive errors, particularly for objects with small dimensions, such as insulators. To mitigate these challenges, and considering the characteristics of the analyzed infrastructure, a method was proposed to suggest regions of interest (ROIs) for the second processing step.

The objects in the "small elements" group—insulators, power disconnectors, and power switches—are found exclusively on support structures such as cantilevers, poles, or gates, which are detected during the first stage of processing. To leverage this observation, a mechanism was developed to group the detected objects from the "mount", "pole", and "horizontal bar" classes using the density-based spatial clustering of applications with noise method (DBSCAN)[29] and to determine their bounding rectangles. The DBSCAN method utilizes a custom distance matrix, defined in Eq. (1), which represents the relative distance between the centers of two bounding boxes (described as x_i, y_i and x_j, y_j), relative to the sum of their height (h_i, h_j) and width (w_i, w_j). By using this form of scaling, it is possible to compare distances between pairs of objects with significantly different bounding box dimensions. The ROI coordinates derived through this process are then used to extract areas of interest for the second processing step (Fig. 4). The resulting clusters of detected objects are referred to as "traction groups".

$$d_{ij} = \sqrt{\left(\frac{x_i - x_j}{w_i + w_j}\right)^2 + \left(\frac{y_i - y_j}{h_i + h_j}\right)^2} \tag{1}$$

Dynamic confidence score threshold (DCST) and ROI masking

In order to increase the precision of inference by reducing the number of false-positive detections, a dynamic confidence score threshold (DCST) was implemented for objects detected during the first processing. The system allows filtering predictions by taking into account both their confidence scores and their positions by linking the value of the confidence score threshold (CST) to the normalized distance from the tracks (d). The distance is calculated as the Euclidean distance in pixels from the track mask and normalized to a range of 0 to 1, where 1 corresponds to the maximum possible Euclidean distance between any two pixels in an image with the given dimensions. The CST value is described by the equation (Eq. 2). Values range from CS_{min}, set as the confidence threshold for the model (working point), to CS_{max}, set as 1.0.

$$CST(d) = \begin{cases} CS_{min} & \text{if } d < d_t^1 \\ \frac{d - d_t^1}{d_t^2 - d_t^1} \times (CS_{max} - CS_{min}) + CS_{min} & \text{if } d_t^1 \le d \le d_t^2 \\ CS_{max} & \text{otherwise} \end{cases} \tag{2}$$

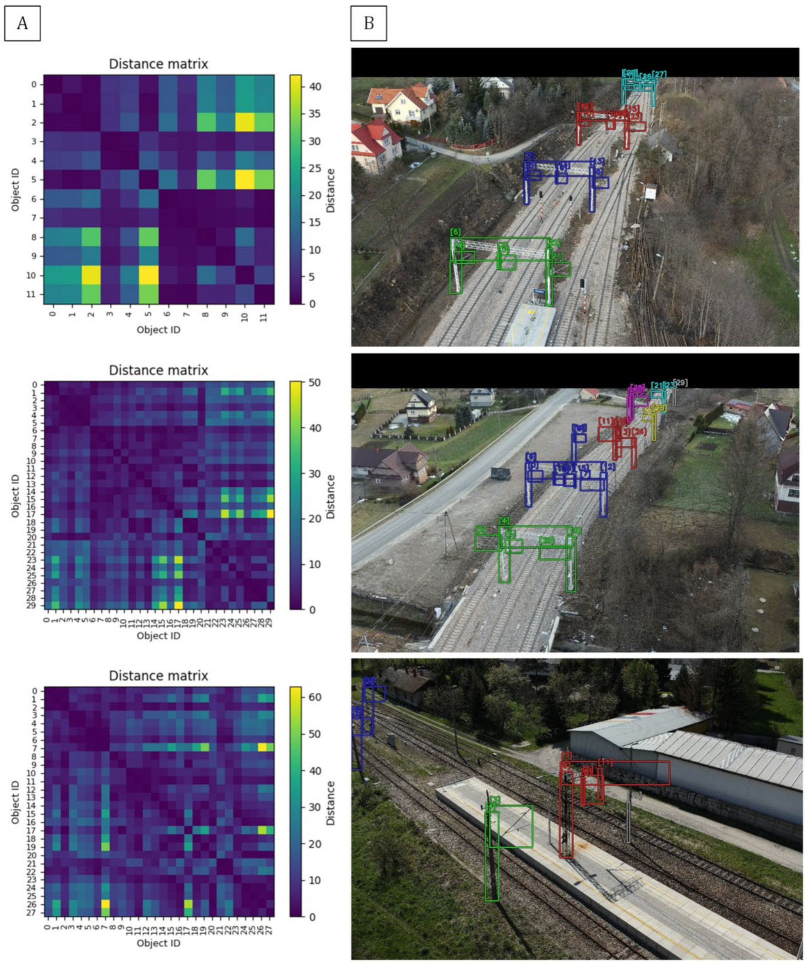

Figure 4. Traction group generation: (**A**) distance matrix between objects, (**B**) visualization of clustering results—unassigned objects are marked in gray, while designated traction groups are highlighted in other colors. *(original images courtesy of GISonLine)*.

The values of the threshold points for the distance from the tracks (d_t^1 and d_t^2) were determined by analyzing the inference results of the system (in two-stage processing mode) on the training dataset. The distances of individual predictions from the tracks, their confidence scores, and their correctness (whether they are false positives or true positives) were evaluated. The distance thresholds were selected using the grid search method, maximizing the objective function that is a weighted harmonic mean between the area under the $CST(d)$ curve (the excluded area) and recall (Eq. 3). Recall is calculated after excluding some predictions according to the determined confidence score thresholds (Eq. 4). The weights were determined empirically and were set to $w_A = 0.175$ and $w_R = 0.825$.

This metric ensures the retention of a high recall while simultaneously selecting thresholds that maximize the excluded area, thereby reducing potential false-positive predictions (increasing precision of the system).

$$F(d_t^1, d_t^2) = \frac{1}{\frac{w_A}{Area} + \frac{w_R}{Recall}} \tag{3}$$

$$Recall = \frac{TP}{TP + FN} = \frac{TP_{initial} - TP_{excluded}}{(TP_{initial} - TP_{excluded}) + (FN_{initial} + TP_{excluded})} \tag{4}$$

The selected threshold values and a visualization of the prediction filtering process is shown in Fig. 5.

The process of determining the weighting matrix begins by scaling the input image to a resolution of 1024x682 and processing it with the track segmentation model. The resulting masks are utilized to create a 512x352 distance map for the track (Fig. 6). For each detection, the corresponding value from the distance map is taken, where the base point is defined as the centre of the bottom edge of the bounding box. The bottom edge was chosen as a reference because it approximates the point of contact between the detected object and the plane on which the tracks are located.

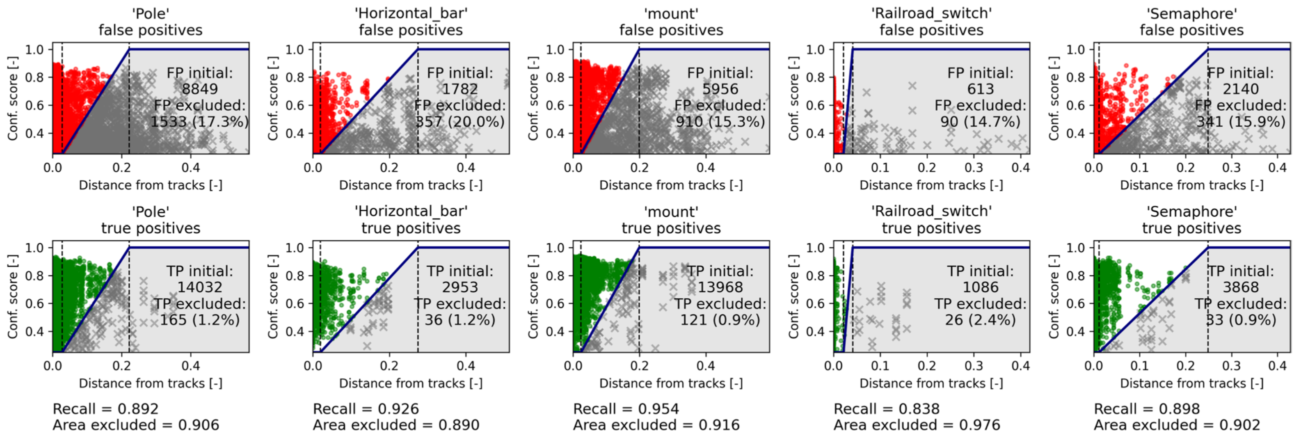

Figure 5. Calculated confidence score threshold curve (blue line).

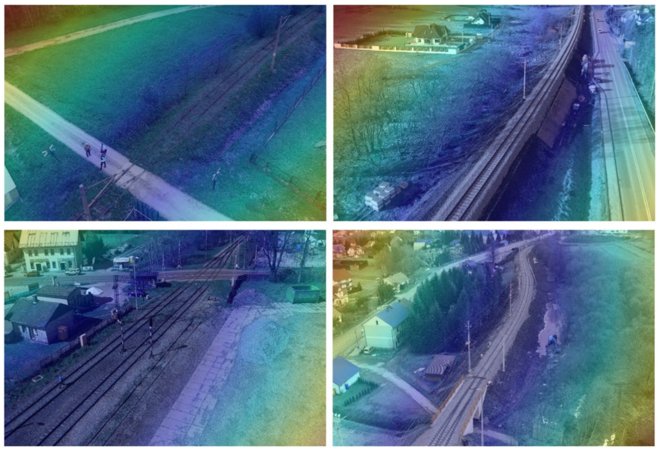

Figure 6. Visualization of track distance maps on input images. (original images courtesy of GISonLine).

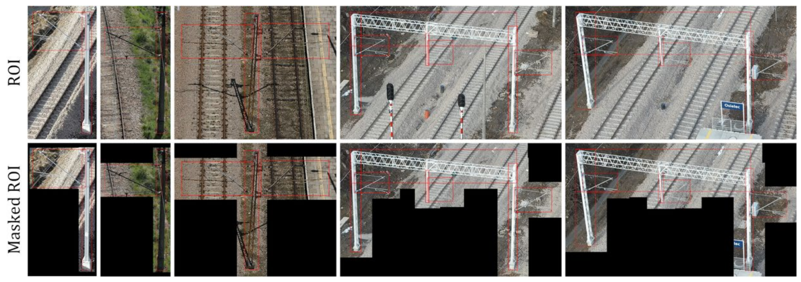

Figure 7. Regions of interest for the second processing stage before and after masking (original images courtesy of GISonLine).

Track segmentation was performed using the YOLO11n (nano) instance segmentation model[30], trained on 86 images with a resolution of 1024x682. The model achieved an AP_{50} of 99.2 and an F_1 score of 98.0 at a confidence threshold of 0.67. This model was used consistently throughout the experiments.

The second stage of detection employs an alternative false-positive reduction method: ROI masking (Fig. 7). This method leverages domain knowledge about the spatial positioning of insulators and other electrical components, specifically that they occur only on support structures or cantilevers. By masking regions outside the bounding boxes of objects belonging to these classes, the method effectively reduces false positives.

Ensembled processing methods

With the ensemble approach, five models were employed simultaneously for the detection task. For each input image tile, five sets of results were accumulated and processed separately for each class. A zero array with the shape of the input image was created, and for each detection, values of 1 were added to it at the detection locations, creating an array of detection intensities with values ranging from 0 to 5. Subsequently, the array was thresholded, retaining only regions (predictions) that were detected more than or equal to the specified value (later refered to as *detection threshold*). The remaining detections were merged and described using an external bounding box (Fig. 8). New confidence score values were calculated as a mean of cumulated detections.

The ensembled method requires the preparation of multiple models trained on different datasets. As an alternative, the authors proposed the Test Time Augmentation (TTA) approach, where a single model is used to perform inference multiple times on slightly modified images. The default implementation of TTA in the Ultralytics framework applies rescale and flip augmentations. Three inferences are performed with the following parameters: (1) scale = 1.0, no flip (original image); (2) scale = 0.83, horizontal flip; (3) scale = 0.67, no flip. The applied image scaling introduces significant changes to the characteristics of the image, which can negatively impact the inference for the problem under consideration. To address this, a custom TTA method was proposed, performing two predictions - one for the original image and another for the horizontally flipped image.

Evaluation measures

Evaluation was performed using *PyCoCoTools* and a test set of 146 high-resolution images. Base confidence threshold (working point) was set to 0.25. AP_{50}, as well as precision, recall, and F_1 metrics were determined, as shown below (Eqs. 5–7):

$$Precision = \frac{TP}{TP + FP}, \tag{5}$$

$$Recall = \frac{TP}{TP + FN}, \tag{6}$$

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}, \tag{7}$$

where TP, FP, and FN represent the number of true positive, false positive, and false negative predictions, respectively, and are calculated as shown in Eq. 8 for all ($N = 146$) test images.

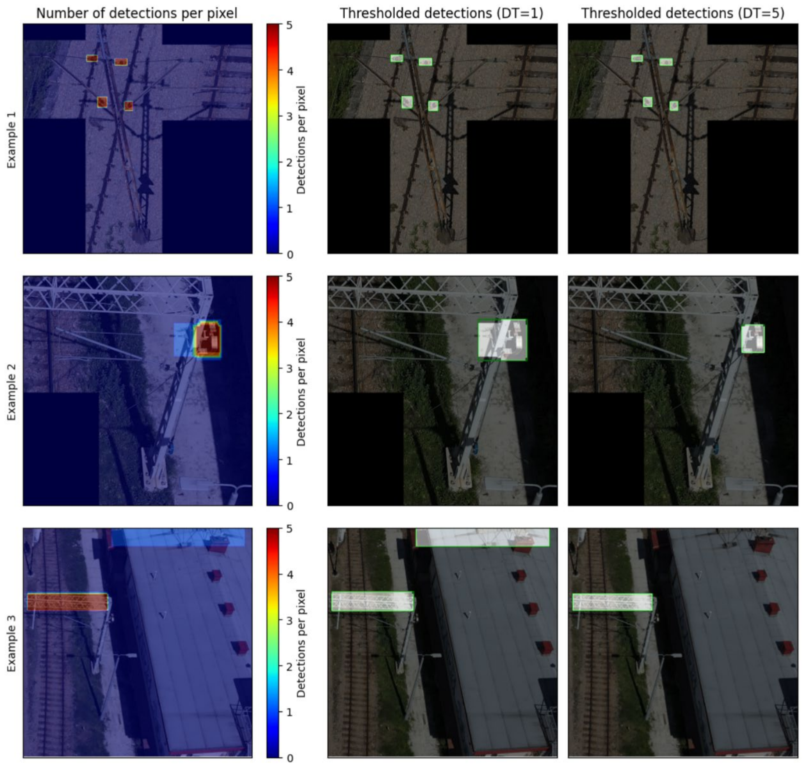
Number of detections per pixel
Thresholded detections (DT=1)
Thresholded detections (DT=5)
Example 1
Example 2
Example 3
Detections per pixel
0
1
2
3
4
5

Figure 8. Detection combination process for different threshold values (DT = 1 and DT = 5) (original images courtesy of GISonLine).

$$\begin{aligned} TP &= \sum_{n=1}^{N} TP_i^{image} \\ FP &= \sum_{n=1}^{N} FP_i^{image} \\ FN &= \sum_{n=1}^{N} FN_i^{image} \end{aligned} \tag{8}$$

Experimental setup

The experimentation, including the training of the machine learning models as well as the validation of the system, was carried out locally on a workstation equipped with an Intel Xeon Silver 4110 CPU (Intel Corporation, USA), 64GB RAM and an Nvidia RTX 4080 graphics card with 16GB VRAM (NVIDIA Corporation, USA). Python programming language together with Pytorch and Ultralytics frameworks were used to prepare the scripts.

Model preparation

Three sets of models were prepared for this study, each trained using a subset of the training dataset derived from different input datasets:

- all elements,
- basic elements,
- small elements.

The models were developed using the five-fold cross-validation methodology, resulting in five distinct models.

In this study, two object detection model architectures were evaluated: YOLO11 (a lightweight convolutional model)[30] and RT-DETR (Real-Time Detection Transformer, a transformer-based model)[31].

In the experimental phase, models with reduced size ("small") were utilized to minimize the time needed for their training and to validate the system with various processing methods. These models were trained for 200 epochs in the "basic elements" and "all elements" scenarios, and for 250 epochs in the "small elements" scenario, using the *Adam* optimizer. For YOLO11s, a learning rate of 1.5×10^{-3} and a batch size of 20 were employed. In the case of RT-DETR-l models, training was conducted with a batch size of 10 and a learning rate of 5×10^{-4}. These training configurations ensured stable convergence for both model types.

For production deployment, models based on the YOLO11l (large) architecture were employed. These models were trained on the full dataset for 500 epochs with a higher learning rate of 5.0×10^{-3} for the first stage and 5.0×10^{-3} for the second stage. These parameters ensured full convergence when training on a larger and more complex dataset.

Evaluation of processing methods

Six processing methods (Table 3) were explored in this study. Each method was tested five times on the same test dataset of 146 images, with each iteration using a different model from the same group. This approach produced five sets of results for each method, enabling the calculation of the mean and standard deviation of metrics for each processing approach.

Examination of methods enhanced by ensembled processing

In addition, the following methods for extending the base approaches were explored for a variable threshold value of the minimum number of detections:

1. Default YOLO TTA,
2. Custom TTA with horizontally flipped images—detections threshold of 2,
3. Multi-model inference (ensembled)—detections threshold in range of 1 to 5.

For the multi-model inference method, five 'basic element' and 'small element' models were used for joint inference, producing single metric values. In the case of dual inference, the same models were utilized to obtain a set of results, for which the mean and standard deviation values were calculated and used for comparison. Both approaches employed two-stage processing with ROI masking and confidence score weighting. The obtained results were compared against the baseline approach without ensemble methods.

Evaluation of domain shift robustness

To assess the generalization capability of the proposed domain-enhanced method and its robustness to domain shift phenomena, a cross-domain validation experiment was proposed using selected domains from the input

ID	Base processing method	Additional processing method	Model 1 type	Model 2 type
1	Simple one-stage	–	All	–
2	Simple two-stage (tiling)	–	Basic	Small
3	Two-stage processing (object grouping)	–	Basic	Small
3+M	Same as 3	Second Stage ROI masking	Basic	Small
3+D	Same as 3	Confidence score weighting	Basic	Small
3+M+D	Same as 3	Confidence score weighting and second stage ROI masking	Basic	Small

Table 3. Proposed processing methods.

dataset (as illustrated in Fig. 2). Weak *YOLO11s* models were trained on the *Osielec 1* and *Osielec 2* domains, which consist of winter aerial images of a modernized station and a section of railway infrastructure. Using a 5-fold cross-validation scheme, five models were trained and subsequently evaluated on images from separate test (target) domains. The experiment was conducted for two approaches: the baseline method (Method 1) and the domain-enhanced method (Method 3+M+D).

The target domains selected for validation were "Osielec 3", "Poronin", and "Jeleśnia". "Osielec 3" corresponds to the same geographical area as the training domains but was captured during the summer, introducing a domain shift primarily due to increased vegetation coverage. In contrast, "Jeleśnia" and "Poronin" represent entirely different locations, with images acquired in late spring. The "Jeleśnia" flight was conducted over a partially modernized station and railway line, while "Poronin" covered a non-modernized segment of the railway. In both cases, the railway infrastructure—specifically the tracks and overhead catenary systems—is of an older type, and the vegetation levels are noticeably higher in "Jeleśnia" and significantly higher in "Poronin" compared to the winter scenes used for training.

These domains were selected to evaluate the model's performance under varying degrees of domain shift: a moderate shift in the case of "Osielec 3", and more substantial shifts in the cases of "Jeleśnia" and "Poronin".

Results and discussion

Evaluation of processing methods

The results of the evaluation of the both YOLO11 and RT-DETR based processing methods are presented in Table 4 and visualized in Fig. 10. The data was collected for two groups of objects: "basic" and "small". For the evaluation of the "basic elements" group, an impact on the obtained quality metrics was observed only for methods 1, 2, and 3+D, namely the one-stage, two-stage, and two-stage with confidence score threshold approaches. The remaining methods only affect objects in the "small" group.

For the objects in the "basic" group using YOLO11-based methods, no significant changes were observed in the values of AP_{50} or Recall, with AP_{50} ranging from 88.95 to 89.44 and Recall from 91.45 to 92.12. However, clear improvements were noted in other metrics. The incorporation of an additional confidence score weighting method (Method 3+D) increased system precision from 81.06 (1.80) to 85.60 (1.63) compared to the simple one-stage processing pipeline (Method 1). This enhancement, while maintaining a high Recall, also raised the F_1 score from 85.88 (0.95) to 88.39 (1.26). Analogous results were obtained for RT-DETR-based methods, where the application of additional processing steps (Method 3+M+D) increased precision to 72.26 (3.47) and

	YOLO11 based methods				
	AP50	**Recall**	**Precision**	**F1**	**Time**
	Basic elements				**Full inference**
1	88.95 (1.17)	91.45 (1.12)	81.06 (1.80)	85.88 (0.95)	0.507 (0.016)
2	89.44 (1.64)	92.12 (1.56)	80.93 (2.28)	86.11 (1.56)	1.839 (0.037)
3	89.44 (1.64)	92.12 (1.56)	80.93 (2.28)	86.11 (1.56)	0.703 (0.055)
3+D	89.33 (1.74)	91.45 (1.59)	85.60 (1.63)	88.39 (1.26)	0.823 (0.014)
3+M+D	89.33 (1.74)	91.45 (1.59)	85.60 (1.63)	88.39 (1.26)	0.894 (0.018)
	Small elements				
1	35.97 (2.55)	81.80 (4.12)	22.44 (1.63)	35.10 (2.09)	
2	70.87 (2.21)	90.84 (2.21)	38.40 (2.65)	53.33 (2.61)	
3	68.32 (2.68)	90.42 (1.07)	51.85 (1.50)	65.79 (1.29)	
3+M	69.50 (1.90)	89.29 (1.41)	55.87 (1.95)	68.67 (1.65)	
3+M+D	68.91 (1.72)	88.63 (1.64)	56.58 (1.87)	69.02 (1.66)	
	RT-DETR based methods				
	AP50	**Recall**	**Precision**	**F1**	**Time**
	Basic elements				**Full inference**
1	89.53 (1.33)	92.74 (1.12)	67.83 (3.90)	78.24 (2.83)	0.745 (0.008)
2	90.83 (2.18)	94.30 (1.53)	63.21 (5.24)	75.37 (4.07)	3.948 (0.021)
3	90.83 (2.18)	94.30 (1.53)	63.21 (5.24)	75.37 (4.07)	1.632 (0.161)
3+D	90.73 (2.16)	93.60 (1.59)	72.26 (3.47)	81.41 (2.56)	1.839 (0.071)
3+M+D	90.73 (2.16)	93.60 (1.59)	72.26 (3.47)	81.41 (2.56)	1.731 (0.040)
	Small elements				
1	42.43 (2.84)	89.28 (3.48)	18.21 (1.21)	30.23 (1.82)	
2	80.82 (2.29)	92.95 (2.20)	26.41 (4.02)	40.39 (5.06)	
3	80.14 (2.81)	93.78 (1.10)	40.61 (4.59)	56.19 (4.71)	
3+M	77.88 (2.40)	92.18 (1.76)	43.61 (3.36)	58.73 (3.36)	
3+M+D	77.50 (2.35)	91.63 (1.79)	45.73 (2.58)	60.60 (2.56)	

Table 4. Results for evaluation of processing methods for YOLO11 and RT-DETR based models.

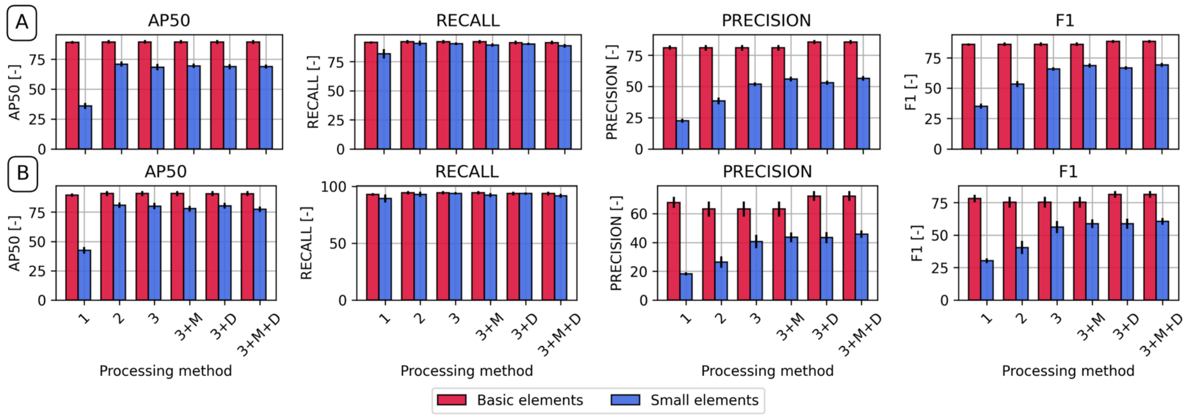

Figure 10. Results for evaluation of (**a**) YOLO11 and (**b**) RT-DETR based processing methods.

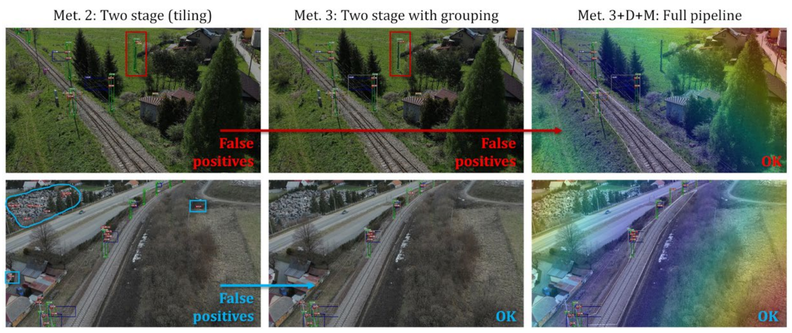
Met. 2: Two stage (tiling)
Met. 3: Two stage with grouping
Met. 3+D+M: Full pipeline
False positives
False positives
OK
False positives
OK
OK

Figure 9. Visualisation of system results for the three processing methods. False-positive errors can be seen for objects in the basic elements (red) and small elements (blue) groups (original images courtesy of GISonLine).

maintained a similar Recall, resulting in an F_1 score of 81.41 (1.66), compared to 78.24 (2.83) obtained with the basic approach.

The results for objects in the "small elements" group clearly demonstrate the effectiveness of the proposed two-stage processing approach. Transitioning from Method 1 to Method 3 resulted in a substantial increase in AP_{50} from 35.97 (2.55) to 68.32 (2.68) for YOLO11. Recall improved from 81.80 (4.12) to 90.42 (1.07), while the most significant gain was observed in precision, which increased from 22.44 (1.63) to 51.85 (1.50). As a result, the F_1 score for Method 3 reached 65.79 (1.29). Applying ROI masking before the second-stage model inference (Method 3+M) further improved precision to 55.87 (1.95), raising the F_1 score to 68.67 (1.65).

Comparable trends were observed for the RT-DETR architecture. The use of Method 3 led to a marked increase in AP_{50} from 42.43 (2.84) to 80.14 (2.81), while recall improved from 89.28 (3.48) to 93.78 (1.10). Precision also saw a notable rise from 18.21 (1.21) to 43.61 (3.36), resulting in an F_1 score of 56.19 (4.71). Incorporating ROI masking prior to second-stage inference (Method 3+M) further improved the results, maintaining the same precision value of 43.61 (3.36) but increasing the F_1 score to 58.73 (3.36). Additional post-processing in the form of confidence score weighting (Method 3+M+D) did not yield significant further improvements for either model.

Another important aspect to consider is the image processing time. The shortest average processing times were recorded for the single-stage method (Method 1), with 0.507 (0.016) seconds for YOLO11 and 0.745 (0.008) seconds for RT-DETR. However, this time efficiency came at the expense of significantly reduced inference quality. In contrast, the simple two-stage method (Method 2), which achieved strong performance in terms of quality metrics, resulted in the longest processing times—1.839 (0.037) seconds for YOLO11 and 3.948 (0.021) seconds for RT-DETR. To address this, object grouping techniques were introduced in Methods 3x, which substantially reduced inference times. Consequently, even for the most complex methodology (Method 3+M+D), processing times remained well below those of Method 2, reaching 0.894 (0.018) seconds per image for YOLO11 and 1.731 (0.040) seconds for RT-DETR.

Considering both the qualitative performance for "basic" and "small" element groups and the improved processing efficiency, Method 3+M+D was selected for the further experimental analyses presented in this study. A visual comparison of selected methods is provided in Fig. 9.

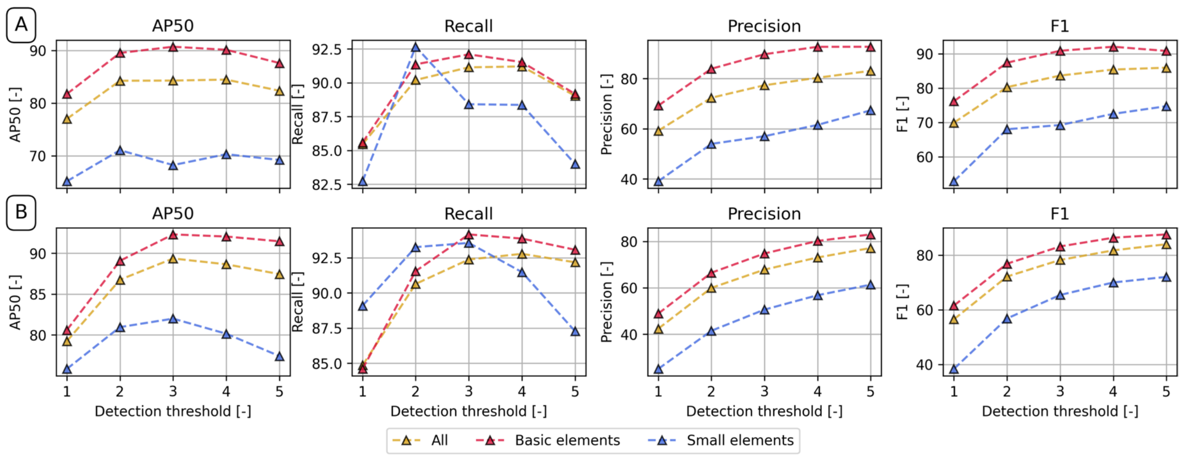

Figure 11. Results for evaluation of ensembled inference with different detections threshold values for (**a**) YOLO11 and (**b**) RT-DETR based processing methods.

	YOLO11 based methods				
	AP50	**Recall**	**Precision**	**F1**	**Time**
	Basic elements				**Full inference**
Base method	89.26 (1.68)	91.45 (1.61)	85.55 (1.61)	88.36 (1.28)	0.894 (0.018)
Default TTA	86.68 (1.64)	89.64 (1.35)	80.33 (1.93)	84.64 (1.32)	1.483 (0.112)
Custom TTA	88.82 (1.36)	90.63 (1.32)	89.67 (1.00)	90.13 (0.77)	1.646 (0.076)
Ensembled (T=4)	90.12	91.52	92.66	92.06	3.195
	Small elements				
Base method	69.24 (1.88)	88.94 (1.35)	56.66 (1.78)	69.17 (1.48)	
Default TTA	69.58 (2.33)	89.55 (0.68)	50.40 (2.21)	64.30 (1.92)	
Custom TTA	68.88 (1.43)	86.75 (1.37)	60.10 (3.22)	70.89 (2.22)	
Ensembled (T=4)	70.30	88.36	61.55	72.53	
	RT-DETR based methods				
	AP50	**Recall**	**Precision**	**F1**	**Time**
	Basic elements				**Full inference**
Base method	90.73 (2.16)	93.60 (1.59)	72.26 (3.47)	81.41 (2.56)	1.731 (0.040)
Default TTA	87.65 (2.22)	90.59 (1.74)	64.76 (3.95)	75.26 (3.13)	3.144 (0.063)
Custom TTA	90.57 (1.12)	92.85 (0.84)	77.38 (3.04)	84.26 (1.97)	2.946 (0.070)
Ensembled (T=4)	92.04	93.86	80.17	86.35	6.165
	Small elements				
Base method	77.50 (2.35)	91.63 (1.79)	45.73 (2.58)	60.60 (2.56)	
Default TTA	77.63 (2.28)	90.91 (1.18)	39.40 (2.81)	54.38 (3.01)	
Custom TTA	79.10 (1.81)	90.14 (1.43)	51.37 (2.40)	65.11 (1.88)	
Ensembled (T=4)	80.14	91.48	56.81	70.09	

Table 5. Comparison between base method (Method 3+M+D) and ensembled variants.

Examination of methods enhanced by ensembled processing

In this approach, the relationship between inference quality metrics and the detection threshold value was first examined, with the results presented in Fig. 11. The analysis was conducted separately for the "basic" and "small" object groups. A detection threshold of 4 was selected for subsequent experiments involving ensembled inference, as it yielded both high precision and recall across both groups and architectures. In contrast, for the TTA method with dual inference, a threshold of 2 was used. The results of the evaluation of the ensembled and TTA processing methods are presented in Table 5 and visualized in Fig. 12.

Among the compared methods, the default TTA approach surprisingly demonstrated the worst metrics. Its application reduced the F_1 score for the "basic elements" group to 84.64 (1.32), primarily due to a significant drop in precision to 80.33 (1.93). For the "small elements" group, the F_1 score fell to 64.30 (1.921), with recall at 89.55 (0.68) and precision at 50.40 (2.21). Additionally, the processing time increased to 1.483 (0.112) seconds. These results unequivocally exclude the possibility of applying this method in industrial settings.

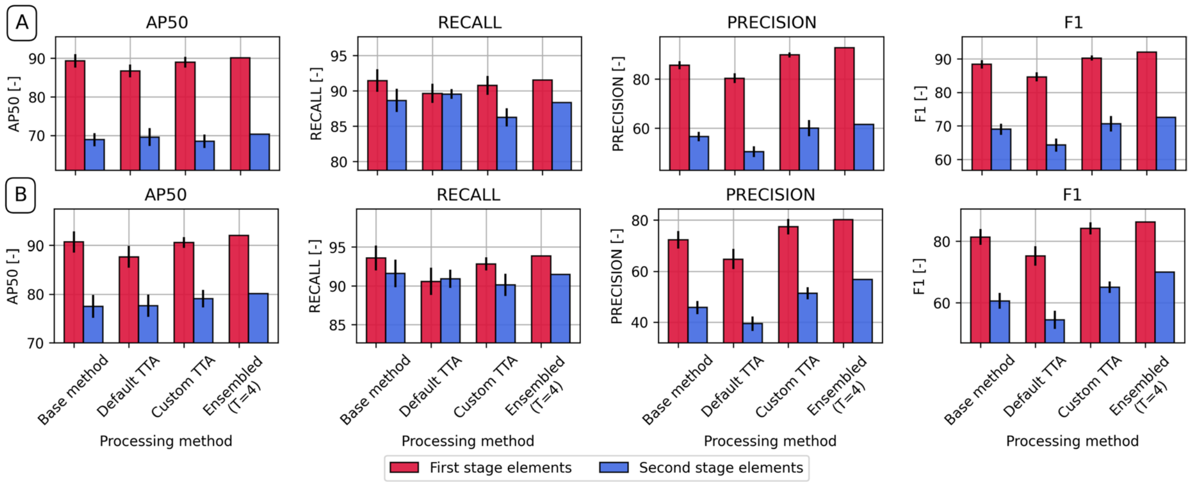

Figure 12. Comparison between base method (Method 3+M+D) and ensembled variants for (**a**) YOLO11 and (**b**) RT-DETR based models.

The highest metrics were observed for the ensembled method, which maintained recall at 91.52 while increasing precision to 92.66, resulting in an F_1 score of 92.06 for the “basic elements” group. For the “small elements” group, an F_1 score of 72.53 was achieved, maintaining recall at 88.36 while increasing precision to 61.55. However, the quality gains in inference were offset by a threefold increase in processing time, reaching 3.195 s per image.

For the custom TTA method, system sensitivity was maintained, with recall at 90.63 (1.32) and an increase in precision to 89.67 (1.00) for the “basic elements” group. For the “small elements” group, a slight decrease in recall to 86.75 (1.37) and an increase in precision to 60.10 (3.22) were observed. The total processing time rose to 1.791 (0.072) seconds. These modest increases in F_1 scores to 90.13 (0.77) and 70.89 (2.22), combined with a limited rise in processing time that remains lower than the typical two-stage approach (Met. 2) and without requiring the training of multiple models as in the ensembled approach, make the custom TTA method promising and justify its further use in industrial applications.

Analogous results were observed for models based on the RT-DETR architecture. However, in this case, inference times increased substantially—reaching 2.946 (0.070) seconds for the custom TTA approach and up to 6.165 seconds for ensemble inference. Such prolonged processing times significantly limit the applicability of these approaches in industrial environments, particularly in scenarios involving the processing of large batches of images acquired during UAV inspection flights.

Evaluation of domain shift robustness

The evaluation results on selected target domains for YOLO11-based models are summarized in Table 6 and visualized in Fig. 13. Two approaches were compared: Method 1 (a simple single-stage processing pipeline) and Method 3+M+D (a domain-enhanced, multi-stage processing approach). The results are presented separately for two element groups: “basic” and “small” elements.

For the *basic* element group, the obtained metric values across the test domains vary significantly. In the case of the Osielec 3 and Poronin domains, performance remained similar to or up to 1.5 times higher than the training-domain reference. In contrast, for the Jeleśnia domain, substantial drops in AP_{50}, Recall, and F_1 scores were observed, with values falling to nearly half of the reference. The performance difference between Method 1 and Method 3+M+D in this group is marginal—both approaches yield comparable results across all three domains.

However, a notable improvement is observed for the *small* element group. In this case, the domain-augmented method consistently achieved higher metric values across all test domains.

These results confirm the effectiveness of domain-informed techniques and underscore their robustness to domain shift, demonstrating performance that is comparable to—or in some cases superior to—standard baseline approaches.

Final system evaluation

Due to the shorter inference time, models from the YOLO11 family were selected for the final evaluation of the developed system. When combined with larger variants of these models, the 3+M+D processing pipeline, and the custom TTA method, the system achieved high performance metrics (Table 7) while maintaining a reasonable inference time of 2.17 seconds per image. For the group of small elements, the AP_{50} score was 69.70, while the F_1 score reached 71.00. These values are notably lowered by the challenging-to-detect *“electrical pole component”* class. Future work should prioritize expanding the dataset with additional instances of this class. For critically important insulators, the AP_{50} and F_1 scores were 74.76 and 75.33, respectively. These results are impressive given the inherent challenges of this class, including its small size and low contrast against the background.

Test domain	Method	AP50	Recall	Precision	F1
Basic elements					
Osielec 1+2 (training)	Met.1	52.19 (3.78)	67.48 (3.48)	50.96 (3.05)	54.80 (3.11)
	Met.3+D+M	51.22 (4.17)	66.79 (3.96)	53.61 (4.13)	56.94 (3.22)
Osielec 3	Met.1	56.54 (5.39)	70.34 (5.75)	60.73 (2.93)	64.11 (4.50)
	Met.3+D+M	55.03 (3.78)	67.84 (4.40)	60.65 (4.49)	63.25 (3.56)
Jeleśnia	Met.1	22.35 (3.80)	29.63 (4.70)	45.74 (14.65)	30.22 (4.85)
	Met.3+D+M	19.44 (2.78)	24.44 (2.98)	48.13 (14.17)	28.56 (3.21)
Poronin	Met.1	59.14 (10.10)	68.74 (10.58)	61.83 (5.56)	64.76 (7.11)
	Met.3+D+M	63.01 (6.37)	66.61 (5.60)	83.13 (8.91)	73.43 (4.96)
Small elements					
Osielec 1+2 (training)	Met.1	23.27 (3.57)	47.88 (6.98)	20.48 (2.76)	27.76 (3.04)
	Met.3+D+M	66.72 (5.35)	83.39 (3.50)	41.40 (5.45)	55.13 (5.59)
Osielec 3	Met.1	16.42 (2.13)	34.07 (4.20)	27.32 (3.58)	25.96 (3.68)
	Met.3+D+M	57.92 (4.18)	76.85 (4.56)	52.51 (2.99)	62.09 (3.17)
Jeleśnia	Met.1	7.91 (1.68)	14.96 (2.76)	20.50 (11.30)	15.86 (2.75)
	Met.3+D+M	19.52 (2.91)	22.86 (3.27)	64.11 (14.45)	30.56 (2.93)
Poronin	Met.1	11.94 (2.61)	21.50 (5.88)	18.79 (2.44)	19.24 (1.81)
	Met.3+D+M	28.66 (4.43)	29.25 (4.41)	47.36 (1.63)	35.99 (3.61)

Table 6. Evaluation results across different target domains, illustrating performance under varying levels of domain shift.

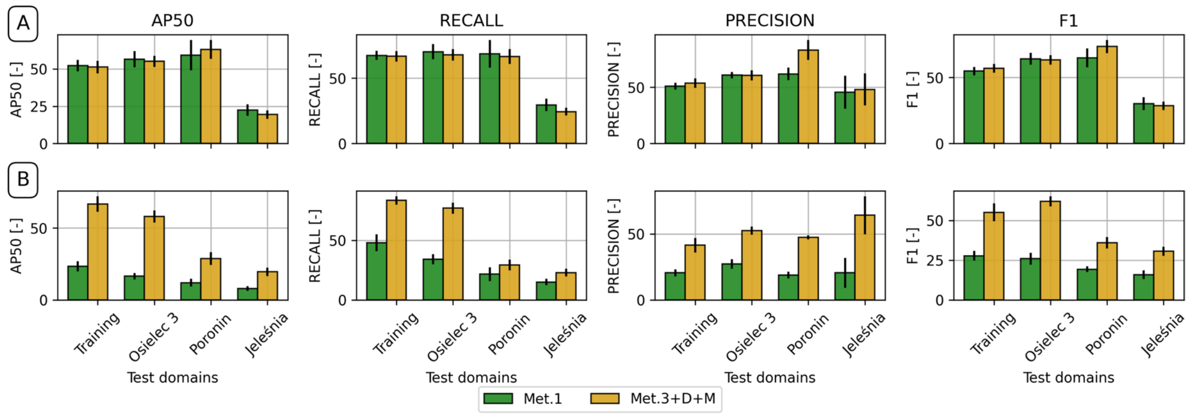

Figure 13. Evaluation results across different target domains, illustrating model performance under varying levels of domain shift: (**a**) basic elements, (**b**) small elements.

Class	AP50	Precision	Recall	F1
Mount	91.13	93.40	92.75	93.07
Semaphore	92.09	89.20	93.14	91.13
Pole	84.58	84.24	88.00	86.08
Horizontal bar	93.62	92.59	96.15	94.34
Railroad switch	92.85	91.57	93.83	92.68
Insulator	74.76	64.54	90.45	75.33
Electrical component	64.66	55.88	82.61	66.67
Group: basic elements	**90.86**	**90.42**	**92.77**	**91.57**
Group: small elements	**69.70**	**60.21**	**86.53**	**71.00**
All classes	**84.82**	**81.79**	**90.99**	**85.70**

Table 7. Results of final system evaluation. Significant values are in bold.

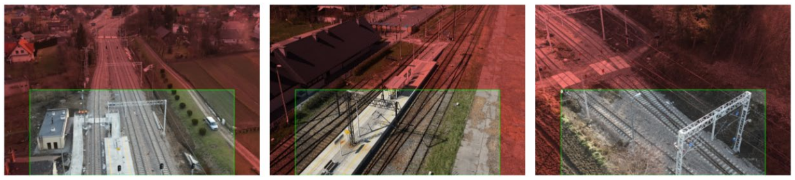

Figure 14. Example images with foreground area highlighted (original images courtesy of GISonLine).

Class	AP50	Precision	Recall	F1
Insulator	88.24	88.44	92.42	90.39
Electrical component	82.39	81.25	86.67	83.87
Group: small elements	**85.35**	**84.85**	**89.59**	**87.13**

Table 8. Results of final system evaluation for foreground elements. Significant values are in bold.

Analysis of the data indicates that the most visible objects are located in the foreground, specifically in the lower half of the image. Considering the nature of the data acquisition process—photos being captured repeatedly during flyovers—it is reasonable to assume that all objects appear at least once in this well-visible region. Based on this, an evaluation of inference metrics for small elements was performed again this time for the foreground area—lower 50% of the image (Fig. 14). Results of this experiment are shown in Table 8. As demonstrated, the values obtained for both classes at this configuration are significantly higher. For the challenging class *"electrical pole component"*, the achieved AP_{50} was 82.39, with an F_1 score of 83.87. Similarly, for insulators, the respective values were 88.24 and 90.39. Considering the previously described data acquisition method and these results, the applicability of the approach outlined in this article for railway infrastructure inspection tasks becomes evident.

Conclusions
This work presents a comprehensive multi-stage, domain-driven framework that leverages domain knowledge to enhance machine learning model inference in railway infrastructure inspection. Specifically, the authors propose five methods:

1. A two-stage processing approach that utilizes the results from the first model to construct "traction groups" and define regions of interest (ROI) for the second machine learning model.
2. Filtering detections based on their distance from the tracks.
3. Masking the input images for the second-stage model using detections from the first stage (ROI masking).
4. A custom test-time augmentation (TTA) method tailored to the problem at hand.
5. An ensemble learning approach employing multiple models.

Together, these methods form a cohesive suite of domain-driven enhancements designed to improve inference quality in railway inspection tasks.

The approaches proposed by the authors provide an additional layer of robustness and optimization, ensuring broader applicability across diverse railway inspection workflows. Moreover, as demonstrated in the conducted experiments, the developed solutions can be effectively applied to models with different architectures (such as YOLO11 and RT-DETR), offering significant enhancements in detection metrics without the need for additional training data.

Presented results highlight the importance of preprocessing strategies, dynamic inference adjustments, and hybrid methodologies in developing scalable, high-performance systems for critical infrastructure monitoring. By applying the proposed methods, the developed processing system achieved an average precision AP_{50} of 84.82 and an F_1-score of 85.70, with an inference time of 2.17 seconds, meeting practical runtime constraints for large-scale, real-world railway inspections.

Although the described methods improve inference quality metrics, they are not a substitute for new data, especially in addressing the domain shift phenomenon. Given the high cost of acquiring new data, legal complications associated with UAV flights over active railway lines, and the significant diversity of railway infrastructure (even within a single country), synthetic data generation emerges as a promising approach. Future work will explore various methods of synthetic data generation, ranging from simple *"copy-and-paste"* techniques to advanced approaches utilizing 3D modeling, rendering environments, and generative models.

In summary, the proposed multi-stage, domain-driven framework demonstrates the value of integrating specialized knowledge into modern ML pipelines for critical infrastructure monitoring. Future research will expand on this foundation by investigating synthetic data generation to further mitigate domain shift, enhance model robustness, and ultimately drive safer, more reliable railway operations. We anticipate that the insights gained here will influence similar inspection challenges in other large-scale, safety-critical infrastructures.

Data availability

The data that support the findings of this study are available from the corresponding author, upon reasonable request.

Received: 26 March 2025; Accepted: 6 August 2025
Published online: 11 August 2025

References

1. Eurostat. Eurostat Database. https://ec.europa.eu/eurostat (2023). Accessed 23 Jan 2025.
2. Peiu, P. & Nemtanu, F. Railways contribution to national security. *Strategies XXI (National Defence College)* **1**, 80–94. https://doi.org/10.53477/2668-5094-21-06 (2021).
3. PKP PLK S.A. Regulamin Sieci 2022/2023 (2021). Accessed 16 Dec 2024.
4. De Ruvo, P., Distante, A., Stella, E. & Marino, F. A GPU-based vision system for real time detection of fastening elements in railway inspection. In *2009 16th IEEE International Conference on Image Processing (ICIP)*. 2333–2336. https://doi.org/10.1109/ICIP.2009.5414438 (2009).
5. Wang, Y. et al. Automatic visual inspection and condition-based maintenance for catenary. In *Maintenance Management*. Chap. 9. https://doi.org/10.5772/intechopen.82149 (IntechOpen, 2019).
6. Huang, Z. et al. Rapid surface damage detection equipment for subway tunnels based on machine vision system. *J. Infrastruct. Syst.* **27**, 04020047. https://doi.org/10.1061/(ASCE)IS.1943-555X.0000591 (2021).
7. Lesiak, P. Inspection and maintenance of railway infrastructure with the use of unmanned aerial vehicles. *Railway Rep.* **188**, 31–43. https://doi.org/10.36137/1883P (2020).
8. Kochan, A., Rutkowska, P. & Wójcik, M. Inspection of the railway infrastructure with the use of unmanned aerial vehicles. *Arch. Transport Syst. Telemat.* **11** (2018).
9. Wu, Y., Qin, Y., Wang, Z. & Jia, L. A uav-based visual inspection method for rail surface defects. *Appl. Sci.* **8**. https://doi.org/10.3390/app8071028 (2018).
10. Bettemir, O. H. Quality and safety assurance of railway tracks by UAV. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 9. 2015 ASME/IEEE International Conference on Mechatronic and Embedded Systems and Applications, V009T07A095. https://doi.org/10.1115/DETC2015-47537 (2015).
11. Ghassoun, Y. et al. Implementation and validation of a high accuracy UAV-photogrammetry based rail track inspection system. *Remote Sens.* **13**. https://doi.org/10.3390/rs13030384 (2021).
12. Kafetzis, D., Fourfouris, I., Argyropoulos, S. & Koutsopoulos, I. Uav-assisted aerial survey of railways using deep learning. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*. 1491–1500. https://doi.org/10.1109/ICUAS48674.2020.9213928 (2020).
13. Mohammad, S. P. *Machine Vision For Automating Visual Inspection Of Wooden Railway Sleepers*. Ph.D. thesis, Dalarna University (2008).
14. Banić, M., Miltenović, A., Pavlović, M. & Ćirić, I. Intelligent machine vision based railway infrastructure inspection and monitoring using UAV. *Facta Univ. Ser. Mech. Eng.* **17**, 357. https://doi.org/10.22190/fume190507041b (2019).
15. Han, Y. et al. Computer vision-based automatic rod-insulator defect detection in high-speed railway catenary system. *Int. J. Adv. Robot. Syst.* **15**, 1729881418773943. https://doi.org/10.1177/1729881418773943 (2018).
16. Felzenszwalb, P. F., Girshick, R. B., McAllester, D. & Ramanan, D. Object detection with discriminatively trained part-based models. *IEEE Trans. Pattern Anal. Mach. Intell.* **32**, 1627–1645. https://doi.org/10.1109/TPAMI.2009.167 (2010).
17. Kang, G., Gao, S., Yu, L. & Zhang, D. Deep architecture for high-speed railway insulator surface defect detection: Denoising autoencoder with multitask learning. *IEEE Trans. Instrum. Meas.* **68**, 2679–2690. https://doi.org/10.1109/TIM.2018.2868490 (2019).
18. Liu, X. et al. Insulator detection in aerial images based on faster regions with convolutional neural network. In *2018 IEEE 14th International Conference on Control and Automation (ICCA)*. 1082–1086. https://doi.org/10.1109/ICCA.2018.8444172 (2018).
19. Wang, S., Niu, L. & Li, N. Research on image recognition of insulators based on YOLO algorithm. In *2018 International Conference on Power System Technology (POWERCON)*. 3871–3874. https://doi.org/10.1109/POWERCON.2018.8602149 (2018).
20. Chen, B. & Miao, X. Distribution line pole detection and counting based on YOLO using UAV inspection line video. *J. Electr. Eng. Technol.* **15**, 441–448. https://doi.org/10.1007/s42835-019-00230-w (2019).
21. Tho Tran, H., Quan Tran, M., Huy Tran, Q. & Cuong Pham, V. A computer vision system for power transmission line inspection robot. In *2021 International Conference on System Science and Engineering (ICSSE)*. 289–294. https://doi.org/10.1109/ICSSE52999.2021.9538440 (2021).
22. Tong, L. et al. Trirnet: Real-time rail recognition network for UAV-based railway inspection. In *IEEE Transactions on Intelligent Transportation Systems*. 1–17. https://doi.org/10.1109/TITS.2023.3328379 (2023).
23. Prapas, I., Derakhshan, B., Mahdiraji, A. R. & Markl, V. Continuous training and deployment of deep learning models. *Datenbank-Spektrum* **21**, 203–212 (2021).
24. von Rueden, L. et al. Informed machine learning— A taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Trans. Knowl. Data Eng.* **35**, 614–633. https://doi.org/10.1109/TKDE.2021.3079836 (2023).
25. Zhang, W. et al. Data-driven deep learning approach for thrust prediction of solid rocket motors. *Measurement* **225**, 114051. https://doi.org/10.1016/j.measurement.2023.114051 (2024).
26. Li, X., Yu, S., Lei, Y., Li, N. & Yang, B. Dynamic vision-based machinery fault diagnosis with cross-modality feature alignment. *IEEE/CAA J. Autom. Sin.* **11**, 2068–2081. https://doi.org/10.1109/JAS.2024.124470 (2024).
27. Li, X., Zhang, W., Li, X. & Hao, H. Partial domain adaptation in remaining useful life prediction with incomplete target data. *IEEE/ASME Trans. Mech.* **29**, 1903–1913. https://doi.org/10.1109/TMECH.2023.3325538 (2024).
28. Zhang, W., Hao, H. & Zhang, Y. State of charge prediction of lithium-ion batteries for electric aircraft with swin transformer. *IEEE/CAA J. Autom. Sin.* **12**, 645–647. https://doi.org/10.1109/JAS.2023.124020 (2025).
29. Ester, M., Kriegel, H.-P., Sander, J. & Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96. 226–231 (AAAI Press, 1996).
30. Jocher, G., Qiu, J. & Chaurasia, A. Ultralytics YOLO (2023).
31. Lv, W. et al. DETRs beat YOLOs on real-time object detection (2023). arxiv:2304.08069.

Acknowledgements

We would like to express our sincere gratitude to our partners from *GISonLine*: Piotr Piech, Tomasz Gacek, Agnieszka Pogorzelska, and Anna Nowicka for their invaluable support, essential data, and expertise.

Author contributions

KM: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data Curation, Writing - Original Draft, Visualization PM: Conceptualization, Methodology, Software, Validation, Writing - Review & Editing; JR: Conceptualization, Resources, Writing - Review & Editing, Supervision, Project administration, Funding acquisition;

Funding

The work described in this article was funded as part of the project *"Development of GISonLine through the creation and implementation of an innovative service for the inventory and monitoring of catenary systems"* financed by the European Regional Development Fund.

Declarations

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to K.M.