Deep Learning for Semantic Segmentation of Linear Infrastructure from UAV Imagery Using NVIDIA Jetson AGX Orin

Justyna S. Stypułkowska*
Łukasiewicz Research Network – Institute of Aviation, Warsaw, Poland
**Corresponding author: Justyna S. Stypułkowska (justyna.stypulkowska@ilot.lukasiewicz.gov.pl)*

Submitted: 30 Aug 2025 **Accepted:** 04 Dec 2025 **Published:** 31 Mar 2026

Abstract A method for semantic segmentation of RGB images captured by UAVs to detect railway infrastructure elements, including tracks, level crossings, and surrounding vegetation is proposed. The study was conducted at the Łukasiewicz Research Network – Institute of Aviation, where a proprietary, manually annotated UAV RGB dataset was created. Five deep neural network architectures were trained and compared: DeepLabV3+, Feature Pyramid Network (FPN), LinkNet, Pyramid Attention Network (PAN) and X-Unet. These models were chosen for their distinct approaches to semantic segmentation and feature processing. Training was performed on a desktop computer with an NVIDIA GeForce RTX 3080 GPU and tests were made also on an NVIDIA Jetson AGX Orin to assess deployment feasibility under real-time conditions. Experimental results confirm the strong performance of the analyzed models in segmenting railway tracks and surrounding vegetation. FPN achieved the highest scores, followed by X-Unet, DeepLabV3+, LinkNet, and PAN. All models operated reliably on the NVIDIA Jetson AGX Orin edge platform. The proposed solution can support remote monitoring of railway infrastructure and vegetation. It can also be adapted to other applications by adjusting the training dataset and object categories. This research demonstrates the potential of deep learning as a powerful tool for analyzing UAV RGB imagery in engineering and environmental contexts.

Keywords: artificial intelligence, deep learning, semantic segmentation, RGB imaging, UAV, NVIDIA Jetson AGX Orin, railway infrastructure, image analysis, real-time monitoring, DeepLabV3+, FPN, PAN, LinkNet, X-Unet.

1. Introduction

Automation of railway infrastructure monitoring can play a significant role in ensuring safety, optimizing maintenance, and controlling vegetation encroaching on the railway right-of-way. Traditional manual inspection methods, although effective, are time-consuming, costly, and prone to subjective errors. These limitations become even more critical in the context of increasing railway traffic and the need for predictive maintenance strategies [19]. Recent studies highlight that automation not only reduces operational costs but also improves reliability and consistency compared to manual inspections [18].

In response to these challenges, this study introduces a semantic segmentation approach for UAV-acquired RGB images using deep learning algorithms. The research aims to automate the detection of key objects (e.g., railway tracks, level crossings, vegetation) and compare the effectiveness of selected neural network architectures, while assessing their deployment compatibility on an NVIDIA Jetson AGX Orin mounted on

a UAV-based module. This dual focus, accuracy and embedded feasibility, addresses a critical gap in the literature, where most works concentrate on segmentation quality without considering deployment constraints on edge AI platforms [28].

The literature offers various methods for object segmentation and aerial infrastructure analysis. Few studies compare the performance of different deep learning models in this specific context; this research addresses that gap. The work presented was carried out as part of the statutory project of the Łukasiewicz Research Network – Institute of Aviation entitled *Assessment and analysis of the potential use of remote object detection methods and condition identification of critical infrastructure*. The originality of this work lies in its comprehensive evaluation of five architectures under identical conditions and practical deployment on an embedded platform, ensuring findings are both theoretically relevant and applicable in real-world UAV monitoring scenarios [11].

Five modern segmentation architectures were trained: DeepLabV3+ [5, 6, 7], Feature Pyramid Network (FPN) [12, 15, 16, 20], LinkNet [4], Pyramid Attention Network (PAN) [14], and X-Unet [21, 22, 25, 31]. All models were trained on the same training dataset to ensure a fair and reliable comparison. Such a standardized setup is rarely reported, as existing studies often use different datasets or protocols, making direct comparisons difficult [3].

The study also enabled an evaluation of the effectiveness of each architecture under real-world application conditions. Each model was assessed in terms of segmentation quality using the following metrics: Pixel Accuracy, IoU, mean IoU, F1-score, mean F1-score, Precision, and Recall. Additionally, the target model's response time was analyzed during deployment on the NVIDIA Jetson AGX Orin to assess real-time monitoring feasibility. This aspect is crucial for UAV-based systems, where latency affects operational safety and decision-making [1].

Evaluating segmentation quality helps to understand how accurately each model represents railway infrastructure objects, which are crucial for effective track condition monitoring and vegetation overgrowth assessment. Inference speed determines the practical feasibility of deploying the solution on a UAV with NVIDIA Jetson AGX Orin, where information must be delivered in real time. This study helps bridge the gap between algorithmic development and practical implementation in edge computing environments [30].

This dual evaluation approach provides both theoretical insights and practical validation of the proposed solutions. The results also lay the groundwork for future research on optimizing segmentation models for other linear infrastructure types and integrating multimodal data (e.g., LiDAR, thermal imaging) to enhance the monitoring [23].

The following sections examine which architectures achieve the highest segmentation accuracy. Identifying the most precise models enables transferring the solution to other infrastructure monitoring systems or extending it to new object classes.

2. Related works

There is a growing interest in methods that leverage deep neural networks for semantic segmentation of linear infrastructure imagery, including railway infrastructure. Of particular relevance is the use of images acquired from unmanned aerial vehicles (UAVs), which enable rapid and accurate data collection over relatively large and, in some cases, difficult-to-access areas. Recent studies emphasize that UAV-based monitoring significantly improves coverage and operational efficiency compared to traditional ground inspections [1, 19].

The literature describes various methods aimed at detecting railway tracks, safety barriers, and other environmental elements. Most existing approaches utilize U-Net architectures and their modifications for semantic segmentation tasks [22]. Alternatively, architectures such as DeepLabV3+ [7], PSPNet, and Feature Pyramid Networks (FPN) [15] are also employed. In addition, attention-based and transformer-enhanced variants of DeepLabV3+ have recently been proposed to improve feature extraction in complex aerial scenes [2, 28].

Some studies use DeepLabV3+ for railway track segmentation in UAV imagery, often paired with lightweight backbones like MobileNetV2 or MobileNetV3 to enable deployment on resource-constrained platforms. One enhanced DeepLabV3+ variant with MobileNetV3 achieved 97.69% accuracy and 88.93% mean IoU, while maintaining good computational efficiency for edge devices such as NVIDIA Jetson [27]. Similar improvements have been reported for adaptive DeepLab variants optimized for UAV imagery, achieving high IoU scores while reducing inference time [2].

Other researchers describe the use of semantic segmentation techniques for detecting elements such as railway tracks, sleepers, and components of catenary systems (OCS). Their work provides a broad overview of methods addressing both segmentation and damage detection of technical components, employing architectures such as FCN and U-Net, LiDAR data, as well as multimodal approaches. This highlights the growing importance of deep learning in this field [10]. Recent multimodal approaches integrate LiDAR and thermal imaging to enhance detection under adverse conditions, such as fog or low light [24, 26].

Despite the focus of many studies on improving segmentation accuracy, relatively few address the implementation of developed solutions on edge computing devices. This aspect is crucial for real-time operation, for example during UAV flights. Some recent publications have responded to this need, demonstrating that image segmentation models can be deployed on small, resource-constrained devices. One example introduces a lightweight U-Net variant tested on CPUs, GPUs, and FPGAs, achieving real-time performance, with the best results on FPGA-based deployment. Lopez-Montiel et al. [17] proposed JetSeg, a simplified segmentation model optimized for Jetson Xavier, offering lower resource consumption and faster operation than typical alternatives, making it

suitable for embedded systems. Further research confirms that hardware-aware designs and attention-based optimizations can significantly improve inference speed on Jetson platforms [13, 23].

Chen and colleagues [8] described a model that can be deployed on edge devices such as the Jetson Nano. Their system is capable of performing real-time image segmentation at approximately 25 frames per second, without the need to transmit data to a cloud environment. This means that railway track monitoring can be performed directly onboard the UAV. Similar results have been achieved using global–local attention mechanisms for UAV imagery, enabling real-time segmentation with reduced latency [30].

Despite these examples, few studies compare segmentation quality and edge performance across multiple architectures. This work addresses that gap by evaluating five architectures: DeepLabV3+, FPN, LinkNet, PAN, and X-Unet for semantic segmentation of UAV-acquired railway imagery. Another contribution is demonstrating these models on NVIDIA Jetson AGX Orin, combining high accuracy with low computational demand, making the approach viable for real-world infrastructure inspection. Processing speed was sufficient for real-time monitoring. This evaluation helps bridge the gap between algorithmic development and practical deployment, supporting future research on scalable UAV-based monitoring systems [3, 11].

3. Data and methods

3.1. Datasets

For the purposes of the research described in this article, a proprietary dataset of annotated RGB images depicting railway infrastructure and surrounding vegetation was developed. The images were acquired using unmanned aerial vehicles (UAVs) as part of work conducted at the Łukasiewicz Research Network – Institute of Aviation. Images were captured using a Sony ILX-LR1 RGB camera mounted on a DJI Matrice M600 UAV at altitudes between 20 and 120 meters under varying lighting and weather conditions. Each image has a resolution of 5280×3956 pixels, ensuring high detail for infrastructure and vegetation analysis.

The dataset consists of 1885 high-resolution annotated RGB images. The annotation process was carried out manually using the Label Studio environment, employing the `polygon` method, which enables precise object labeling. The annotations were performed by employees of the Łukasiewicz Research Network – Institute of Aviation, including the author of this paper.

Six semantic classes were defined for the annotation process, selected based on their relevance to the analysis of the railway infrastructure environment:

- *railway* – railway tracks and infrastructure elements,
- *trees*,

- *otherplants* – non-arboreal residual vegetation,
- *levelcrossing*,
- *background* – remaining areas irrelevant to the classification process.

The dataset was split into training and validation subsets in a 70 : 30 ratio, maintaining a representative class distribution across both partitions. For final evaluation, the validation subset was reused as the test set. This ensured models never saw test images during training. However, reusing the validation set as the test set does not preserve evaluation integrity, as a dedicated test set is needed for unbiased performance estimation.

Due to confidentiality restrictions imposed by the employer and project agreement, the dataset cannot be made publicly available. Representative sample images included in the training dataset are presented in Fig. 1.

3.2. Methods

The primary research method adopted in this study involved a comparative analysis of the accuracy of selected deep learning architectures in the task of semantic segmentation of RGB images. These images depict elements of linear infrastructure and the surrounding vegetation. The main focus was placed on railway infrastructure, including level crossings and the adjacent vegetation. This task is of particular importance in the context of automating technical inspections. It also concerns detecting areas where vegetation, especially taller forms such as trees, encroaches upon the *railway* right-of-way. Ultimately, such automatic detection is intended to support decision-making processes by railway infrastructure managers. Specifically, it helps in determining the necessity of clearing selected sections of railway lines from excessive overgrowth of tall vegetation.

As part of the conducted study, five deep neural network architectures representing different methodological approaches were compared. The selected models included: X-Unet, DeepLabV3+, Feature Pyramid Network (FPN), LinkNet, and Pyramid Attention Network (PAN). These architectures were chosen to ensure that the study encompassed both models focused on high-precision structural detail reconstruction, such as DeepLabV3+ and PAN, as well as lightweight solutions optimized for real-time performance, such as LinkNet.

All models were trained and evaluated within a unified experimental environment, which ensured consistency in data preprocessing, augmentation strategies, optimization methods, and performance reporting procedures.

Implementation used PyTorch Lightning with the segmentation_models.pytorch library. X-Unet was custom-built based on U-Net++ and nested U-Net concepts, adapted for multi-class semantic segmentation. All trained models were evaluated using standard segmentation quality metrics, including Pixel Accuracy, Precision, Recall, F1-score, and

Fig. 1. Sample images originating from the training dataset.

Intersection over Union (IoU), calculated separately for each class as well as reported as a weighted average across all classes.

All experiments were conducted under uniform conditions to ensure reliable comparison of architectures for UAV imagery analysis. Operational performance was also assessed during deployment on NVIDIA Jetson AGX Orin, including inference time, memory usage, and compliance with platform requirements.

3.2.1. Architectures of the models

This chapter provides an overview of the deep learning architectures investigated in this study, namely: DeepLabV3+, Feature Pyramid Network (FPN), X-Unet, LinkNet, and Pyramid Attention Network (PAN).

DeepLabV3+

The DeepLabV3+ architecture was designed with the goal of achieving high-precision semantic image segmentation. It represents an extension of earlier versions of the DeepLab model developed by Google Research, starting from DeepLabV1, through DeepLabV2 and DeepLabV3, up to DeepLabV3+. This architecture was the first to combine atrous convolutions with a complete decoder mechanism within an encoder-decoder structure [5, 6, 7].

A key feature shared across all versions of DeepLab is the use of atrous convolutions, which enable the expansion of the receptive field of filters without reducing the resolution of the feature maps or increasing the number of parameters. These modifications allow the model to capture broader spatial context, which is crucial for objects with variable scale, especially in UAV imagery captured at different altitudes. In the context of the present study, DeepLabV3+ proves well-suited for the analysis of elements such as trees, railway tracks, and trackside infrastructure.

In DeepLabV3, the Atrous Spatial Pyramid Pooling (ASPP) module was introduced as several parallel convolutional paths with different dilation rates, along with a global average pooling component. ASPP enables simultaneous analysis of local and global features, offering a rich contextual representation [6].

DeepLabV3+ adds a decoder that refines object boundaries by combining deep and shallow features. It uses upsampling, skip connections, and convolutional layers to integrate multi-level information, improving contour delineation, critical for segmenting tracks, crossings, and vegetation [7].

The DeepLabV3+ architecture can be configured with various encoder backbones. In the present study, the configuration employed a ResNet-50 encoder pretrained on the ImageNet dataset.

Feature Pyramid Network (FPN)

The next architecture discussed is the Feature Pyramid Network (FPN). Similar to DeepLabV3+, it is a convolutional architecture designed to effectively represent objects at multiple scales. FPN extends traditional CNNs, which often struggle with detecting objects of varying sizes. Initially introduced for object detection, it is now widely used in semantic segmentation [12, 15]. A distinctive feature of the Feature Pyramid Network (FPN) is its use of a bottom-up pathway and a top-down pathway, interconnected through lateral connections. The bottom-up pathway serves as a deep feature extractor, typically a backbone such as ResNet, which generates successive feature maps at

progressively lower spatial resolutions. The top-down pathway, in turn, performs progressive upsampling of high-level semantic feature maps, which are then merged with lower-level features via lateral connections. Lateral connections use 1×1 convolutions to align channel numbers, enabling efficient feature map summation. In semantic segmentation tasks, FPN is commonly used as a decoder-supporting component that facilitates the propagation of multi-scale feature maps, thereby improving the segmentation accuracy of small objects [16, 20]. This mechanism is crucial for segmenting vegetation, tracks, and crossings, especially when these elements vary in scale and position in UAV imagery. In this study, the FPN architecture was used with a ResNet-50 encoder pretrained on the ImageNet dataset. The model was adapted for the task of multi-class segmentation of RGB images captured by UAVs. The implementation leveraged the segmentation_models.pytorch library, which enabled standardization of the training process and facilitated direct comparison of results across different architectures. FPN is a lightweight architecture that can be deployed on low-power devices such as the NVIDIA Jetson AGX Orin [29]. This makes it a practical balance between computational cost and segmentation quality.

LinkNet

LinkNet is a lightweight encoder–decoder segmentation architecture designed with real-time performance in mind. It was first introduced by Chaurasia and Culurciello [4] as a solution optimized for both speed and segmentation accuracy. Due to its compact design and efficient use of residual connections, LinkNet has been successfully applied in various domains, including autonomous systems, edge devices, and UAV-based applications.

The core structure of LinkNet follows a classical encoder–decoder scheme, with a distinctive feature being the use of shortcut connections between corresponding encoder and decoder blocks. These connections transfer feature maps before downsampling, helping preserve spatial information and reduce loss during network propagation.

One of the key features of the LinkNet architecture is its encoder, which is based on a lightweight ResNet-class model. Each encoder block performs downsampling and feature extraction, with the resulting features passed to the corresponding decoder blocks. In the decoder, each block adds features from its encoder counterpart, applies 3×3 convolutions, and upsamples. By limiting operations and focusing on coarse features, LinkNet achieves high throughput with minimal accuracy loss. The architecture is capable of real-time performance even on edge platforms such as the NVIDIA Jetson TX2 and Jetson AGX Orin.

In this study, the LinkNet architecture was used with a ResNet-50 encoder pretrained on the ImageNet dataset. The network was implemented using the Python library segmentation_models.pytorch [9] to ensure consistent training and evaluation. LinkNet is particularly well-suited for deployment in edge computing systems, such as real-time UAV platforms. It serves as an onboard processing component where inference speed is as important as segmentation accuracy.

Pyramid Attention Network (PAN)

The Pyramid Attention Network (PAN) is a deep learning architecture designed for semantic image segmentation, with a particular focus on accurately capturing object structures and boundaries. The model was proposed by Li et al. in 2018 [14] as an extension of classical approaches such as PSPNet and DeepLab, which, despite their high effectiveness, do not sufficiently account for the spatial selectivity of features.

PAN builds on ResNet-based convolutional networks and extends the decoder for dense prediction tasks [14]. It includes two modules: Feature Pyramid Attention (FPA) and Global Attention Upsample (GAU). FPA processes encoder outputs using parallel convolutions with different kernel sizes and global average pooling to create an attention-enhanced spatial-semantic representation. GAU merges high-level features from FPA with detailed lower-level features, enabling effective context propagation in the decoder.

In this study, PAN was adapted for multi-class segmentation of RGB images acquired by UAVs. The variant used a ResNet-50 encoder pretrained on ImageNet and targeted classes such as *railway*, *trees*, *otherplants*, *levelcrossing*, and *background*. Implementation relied on segmentation_models.pytorch for seamless integration into the standardized training environment.

X-Unet

The X-Unet architecture is an extension of the U-Net architecture, incorporating a nested skip connections mechanism. It draws upon concepts introduced in UNet++ [31] and U-Net [21]. In this study, we employed an open-source implementation available in the lucidrains/x-unet repository, licensed under the MIT License [25].

The U-Net architecture was originally designed to be used in biomedical image segmentation [22]. Extensions like UNet++ [31] introduced dense hierarchical connections between encoder and decoder, improving segmentation performance. X-Unet builds on this by using a deep decoder and feature consolidation to integrate multi-level information.

The model includes a downsampling encoder and an extended upsampling decoder with nested, multi-level structures. Key mechanisms are nested skip connections, feature map consolidation, and modular design. Skip connections propagate features from deeper to higher layers, while consolidation aggregates multiple decoder paths for richer feature maps. The modular design allows adjusting nesting depth and channel count.

Unlike standard U-Net, X-Unet enables hierarchical, multi-stage feature propagation through its nested structure, improving fine detail representation and class separation along boundaries.

In this study, X-Unet was adapted for the task of multiclass segmentation on RGB images acquired from BSP. The version used a ResNet-50 encoder pretrained on ImageNet and was trained in PyTorch under the unified experimental protocol described in Subsection 3.2.

Tab. 1. Hyperparameters and Configuration of the Segmentation Models Used in the Study.

Model	Backbone	LR	Weight Decay	Batch Size	Epochs	Early Stopping	Loss Functions	Scheduler
DeepLabV3+	ResNet-50	0.0001	1e-4	4	100	patience = 10	CrossEntropyLoss DiceLoss	ReduceLROnPlateau
FPN	ResNet-50	0.0001	1e-4	4	100	patience = 10	CrossEntropyLoss DiceLoss	ReduceLROnPlateau
PAN	ResNet-50	0.0001	1e-4	4	100	patience = 10	CrossEntropyLoss DiceLoss	ReduceLROnPlateau
X-Unet	Custom Encoder	0.0001	1e-4	4	100	patience = 10	CrossEntropyLoss DiceLoss	ReduceLROnPlateau
LinkNet	ResNet-50	0.0001	1e-4	4	100	patience = 10	CrossEntropyLoss DiceLoss	ReduceLROnPlateau

3.2.2. Implementation, training and evaluation procedures

All architectures utilized in this study were implemented using the PyTorch library. Each model followed a unified encoder-decoder scheme, with encoders based either on pre-trained ResNet-50 models (as in DeepLabV3+, FPN, PAN, and LinkNet) or on a dedicated convolutional encoder, as in the case of X-Unet. The decoder structure varied depending on the architecture, while the segmentation head was configured to perform classification across five analyzed object classes. For architectures using ResNet-50 encoders, weights were initialized from ImageNet pretraining, while the decoder and segmentation head were initialized randomly. The X-Unet model was trained from scratch with random weight initialization.

Training was performed on a Windows 10 desktop with an NVIDIA GeForce RTX 3080 GPU in a CUDA 11.6-enabled virtual environment. The Adam optimizer was used with an initial learning rate of 0.0001 and weight decay of 0.0001. A ReduceLROnPlateau scheduler halved the learning rate after five epochs without improvement in mean IoU, and early stopping was applied with a patience of 10 epochs.

All evaluated architectures were trained using the same pipeline. RGB images were rescaled and cropped to a resolution of 512×512 pixels and normalized to the $[0, 1]$ range. Data augmentation with Albumentations included 90° rotations, flips, contrast and brightness adjustments, and random cropping. Ground truth masks were encoded as integer label maps corresponding to six classes: *railway*, *trees*, *otherplants*, *levelcrossing*, and *background*.

Each model was trained for up to 100 epochs with a batch size of 2. The loss function combined DiceLoss and CrossEntropyLoss with equal weights (0.5 : 0.5). After each epoch, metrics such as pixel accuracy, IoU, mean IoU, precision, recall, and F1-score were computed. Models were saved based on the best mean IoU (mIoU) on the validation set.

All experiments used identical data splits, loss functions, metrics, and augmentation methods, ensuring an objective comparison of architectures. The hyperparameters and configuration data for segmentation models used are collected in Tab. 1.

Tab. 2. Evaluation Results for Pixel Accuracy, IoU, F1-score, Precision, and Recall for the DeepLabV3+ Architecture.

Class	Pixel Accuracy	IoU	F1-score	Precision	Recall
0	0.894730	0.796607	0.886791	0.878990	0.894730
4	0.942189	0.733874	0.846514	0.768478	0.942189
5	0.780494	0.632202	0.774662	0.768916	0.780494
9	0.931077	0.878396	0.935262	0.939485	0.931077
13	0.806878	0.702645	0.825357	0.844703	0.806878
Mean	0.867882	0.748745	0.853717	0.840115	0.871074

Tab. 3. Evaluation Results for Pixel Accuracy, IoU, F1-score, Precision, and Recall for the FPN Architecture.

Class	Pixel Accuracy	IoU	F1-score	Precision	Recall
0	0.909824	0.828377	0.906134	0.902473	0.909824
4	0.849934	0.795032	0.885814	0.924857	0.849934
5	0.793869	0.671752	0.803650	0.813676	0.793869
9	0.928339	0.886766	0.939985	0.951927	0.928339
13	0.878209	0.773510	0.872293	0.866456	0.878209
Mean	0.892011	0.791087	0.881575	0.891878	0.872035

4. Experimental results

This chapter presents the evaluation metric results for each of the architectures analyzed in the study. Summary tables Tab. 2 through Tab. 6 report the following metrics: Pixel Accuracy, IoU, F1-score, Precision, and Recall for the individual classes: *railway*, *levelcrossing*, *trees*, *otherplants*, and *background*. Additionally, average values of these metrics, aggregated across all classes, are provided. The tables correspond to the following architectures: DeepLabV3+ (Tab. 2), FPN (Tab. 3), LinkNet (Tab. 4), PAN (Tab. 5), and XUnet (Tab. 6).

Tab. 4. Evaluation Results for Pixel Accuracy, IoU, F1-score, Precision, and Recall for the LinkNet Architecture.

Class	Pixel Accuracy	IoU	F1-score	Precision	Recall
0	0.907080	0.785819	0.880066	0.854614	0.907080
4	0.949262	0.728692	0.843056	0.758224	0.949262
5	0.701903	0.596667	0.747391	0.799182	0.701903
9	0.951184	0.855406	0.922069	0.894683	0.951184
13	0.765662	0.688954	0.815835	0.873045	0.765662
Mean	0.858954	0.731107	0.841683	0.835949	0.855018

Tab. 5. Evaluation Results for Pixel Accuracy, IoU, F1-score, Precision, and Recall for the PAN Architecture.

Class	Pixel Accuracy	IoU	F1-score	Precision	Recall
0	0.734340	0.604364	0.753400	0.773476	0.734340
4	0.056374	0.052355	0.099500	0.423390	0.056374
5	0.316605	0.261133	0.414125	0.598463	0.316605
9	0.925309	0.525510	0.688963	0.548789	0.925309
13	0.610132	0.481592	0.650100	0.695673	0.610132
Mean	0.686672	0.384991	0.521218	0.607958	0.528552

Tab. 6. Evaluation Results for Pixel Accuracy, IoU, F1-score, Precision, and Recall for the XUnet Architecture.

Class	Pixel Accuracy	IoU	F1-score	Precision	Recall
0	0.886742	0.810324	0.895225	0.903872	0.886742
4	0.851897	0.793492	0.884857	0.920471	0.851897
5	0.749572	0.631012	0.773767	0.799578	0.749572
9	0.927654	0.876006	0.933905	0.940242	0.927654
13	0.889751	0.743043	0.852581	0.818392	0.889751
Mean	0.877694	0.770775	0.868067	0.876511	0.861123

For detailed analysis, it is advisable to focus on the following metrics: mean IoU, IoU per class, mean F1-score, and F1-score per class.The dataset is imbalanced, with the *levelcrossing* class notably underrepresented. These metrics allow for a more accurate interpretation of results.

The IoU per class metric measures how much of the actual area of a given class is correctly predicted by the model. This metric is crucial for assessing segmentation quality per class, including rare ones like *levelcrossing*. Without it, a high global score could mask poor detection of infrequent classes.

The mean IoU metric, on the other hand, represents the average IoU across all classes. It evaluates segmentation quality uniformly across classes, avoiding bias toward more frequent ones.

F1-score is the harmonic mean of precision and recall for each class. It captures both under- and over-segmentation, making it effective for detecting errors and handling classes with fewer pixels.

Mean F1-score is the average of F1-scores across all classes. It reflects the overall classification effectiveness regardless of class frequency and complements the mean IoU metric.

Table 7 is a summary of IoU metric results for all analyzed classes and for each of the deep learning architectures evaluated in this study. Additionally, the results for the mIoU

Tab. 7. Summary of IoU and Mean IoU Metric Results for Individual Models.

IoU	DeepLabV3+	FPN	LinkNet	PAN	Xunet
Class 0	0.796607	0.828377	0.785819	0.604364	0.810324
Class 4	0.733874	0.795032	0.728692	0.052355	0.793492
Class 5	0.632202	0.671752	0.596667	0.261133	0.631012
Class 9	0.878396	0.886766	0.855406	0.525510	0.876006
Class 13	0.702645	0.773510	0.688954	0.481592	0.743043
Mean IoU	0.748745	0.791087	0.731107	0.384991	0.770775

Tab. 8. Summary of F1-score and Mean F1-score Metric Results for Individual Models.

F1-score	DeepLabV3+	FPN	LinkNet	PAN	Xunet
Class 0	0.886791	0.906134	0.880066	0.753400	0.895225
Class 4	0.846514	0.885814	0.843056	0.099500	0.884857
Class 5	0.774662	0.803650	0.747391	0.414125	0.773767
Class 9	0.935262	0.939985	0.922069	0.688963	0.933905
Class 13	0.825357	0.872293	0.815835	0.650100	0.852581
Mean F1-score	0.853717	0.881575	0.84168	0.521218	0.868067

metric are also presented. As shown, the best performance in both per-class IoU and mIoU was achieved by the Feature Pyramid Network (FPN) architecture, followed by XUnet, DeepLabV3+, LinkNet, and finally Pyramid Attention Network (PAN), which obtained the lowest values for these metrics.

Table 8 presents the F1-score results for all classes and for each of the deep learning architectures evaluated in this study. Additionally, the results for the mean F1-score metric are also included. As can be observed, the best performance in both per-class F1-score and mean F1-score was achieved by the Feature Pyramid Network (FPN) architecture, followed by XUnet, DeepLabV3+, LinkNet, and finally Pyramid Attention Network (PAN), which obtained the lowest values for these metrics.

Figure 2 presents a graphical comparison of the mean IoU and mean F1-score values for the deep learning architectures analyzed in this study.

4.1. Model deployment on NVIDIA Jetson AGX Orin

The trained models were subsequently deployed on an NVIDIA Jetson AGX Orin unit to evaluate their performance in a real-time mission scenario. The deployment was made possible using the PyTorch Lightning library along with other Python libraries, including json, time, pathlib, albumentations, cv2, numpy, shapely, torch, skimage, pyvips, and PIL.

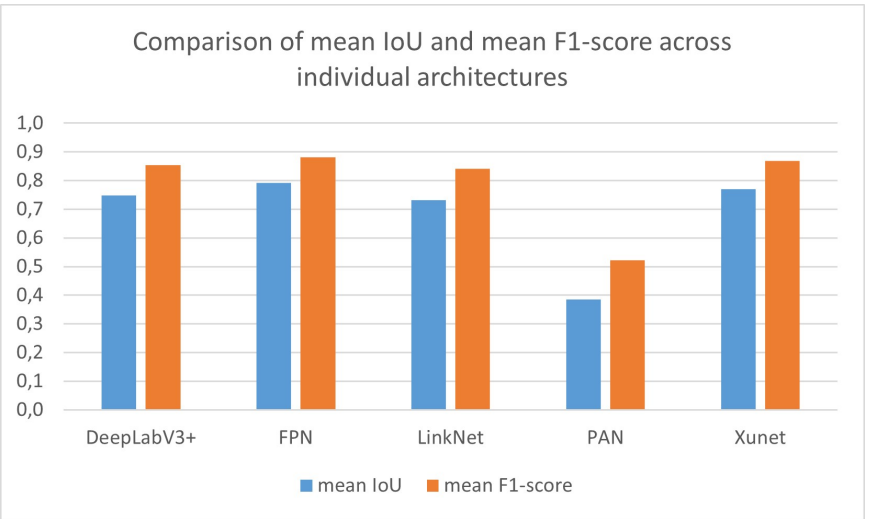

Fig. 2. Comparison of Mean IoU and Mean F1-score Values Across Individual Models.

The runtime code loads the trained model and is capable of analyzing incoming images in sequence. To achieve this, a monitoring mechanism was implemented to watch a designated folder where images from the ongoing mission are continuously saved. For each newly detected image, the system performs semantic segmentation using the classes defined during model training.

The results are generated in three formats: 1° as visualizations in which the predicted masks are overlaid semi-transparently on the original image, which also serves as a background for interpretation; 2° as JSON files saved in two separate directories, one for all analyzed images and another exclusively for alert cases, i.e., when the model detects dangerously close proximity between tall vegetation and railway tracks.

Each JSON file contains detailed information such as: sets of points outlining the contours of detected regions by class, class probability scores, the shortest distances between the *railway* and *trees* classes, flagged alert distances, the file path of the source image, timestamp, prediction time, image dimensions, an alert flag indicating whether a critical distance was detected, and the coordinates of the points that triggered the alert.

The visual and JSON outputs are then used for real-time visualization in a dedicated application designed specifically for this purpose.

All tested architectures were designed and implemented to ensure compatibility with deployment on the NVIDIA Jetson AGX Orin platform. Although inference was executed under conditions characteristic of real-time operation, no quantitative performance metrics (e.g., FPS, power consumption, energy efficiency) were analyzed. Therefore, the

contribution is limited to demonstrating deployment feasibility rather than providing a full performance evaluation. These aspects represent an important direction for future research focused on optimizing embedded inference.

4.2. Model performance results in a real-world field mission

Figure 3 presents a sample of input data in the form of RGB images acquired from a camera mounted on a UAV unit, along with a sample of output data in the form of visual files showing the results produced by the trained model. These visualizations illustrate how the model assigned detected regions to the appropriate classes. According to the adopted color scheme, the *railway* class is marked in green, *otherplants* in purple, and *trees* in pink.

5. Conclusions

The conducted research successfully achieved the key scientific objectives. Five models based on deep learning architectures were trained: DeepLabV3+, FPN, X-Unet, LinkNet, and PAN. The training conditions and parameters were carefully selected to enable a reliable comparison of the individual architectures. The comparison used metrics such as Pixel Accuracy, IoU, F1-score, Precision, and Recall, computed per class and averaged across all classes, with emphasis on mean IoU and mean F1-score as key indicators. FPN achieved the highest performance, followed by X-Unet, DeepLabV3+, LinkNet, and PAN.

The trained models were successfully deployed on the target platform, an NVIDIA Jetson AGX Orin module mounted onboard the DJI Matrice M600 UAV. Deployment required appropriate adaptation of the runtime environment.

The best-performing models accurately captured features in RGB images, correctly delineating object classes with precise contours. Real-time visualizations were generated directly on the Jetson AGX Orin onboard the UAV.

In addition to these findings, the study highlights a trade-off between segmentation accuracy and computational efficiency. FPN achieved the highest accuracy, while X-Unet delivered competitive results with lower complexity, making it suitable for real-time embedded deployment. These insights suggest that architecture selection should reflect application priorities: high accuracy for offline analysis versus lightweight models for UAV-based real-time monitoring. The proposed approach also shows potential for adaptation to other types of linear infrastructure, such as roads or power lines, with appropriate adjustments.

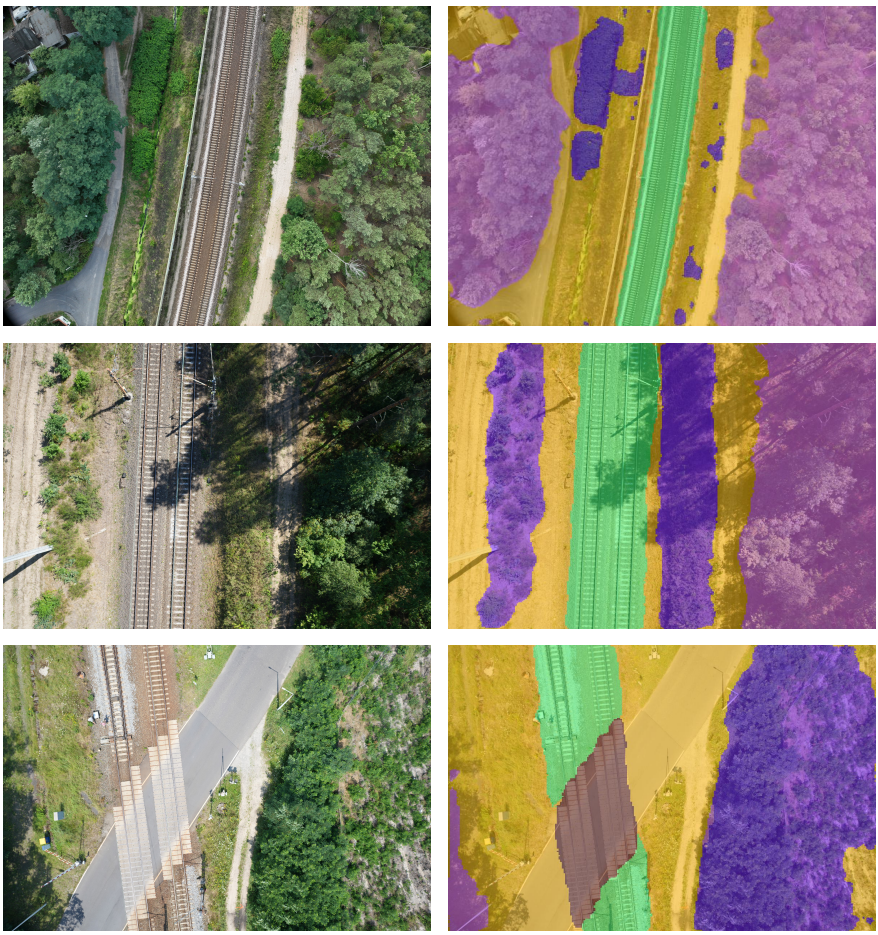

Fig. 3. Input images (on the left) and output results presented as visualizations (on the right). The results pertain to the XUnet architecture.

6. Discussion of limitations

The experimental results revealed differences in performance among the tested architectures. FPN achieved the highest mean IoU (0.791) and mean F1-score (0.882), indicating superior segmentation quality across all classes. This can be attributed to its multi-scale feature aggregation strategy, which effectively captures both global and local context. DeepLabV3+ also delivered strong results (mIoU = 0.749, mean F1-score = 0.854), benefiting from the Atrous Spatial Pyramid Pooling module, though its computational

complexity makes it less suitable for resource-constrained environments. X-Unet demonstrated competitive accuracy (mIoU = 0.771, mean F1-score = 0.868) while maintaining a lightweight structure, which is particularly advantageous for real-time deployment on embedded platforms such as NVIDIA Jetson AGX Orin. LinkNet provided moderate performance (mIoU = 0.731, mean F1-score = 0.842), representing a compromise between accuracy and efficiency. In contrast, PAN obtained the lowest scores (mIoU = 0.385, mean F1-score = 0.521), primarily due to difficulties in segmenting rare classes like level crossings, suggesting that attention-based mechanisms alone may not suffice without robust multi-scale processing. These findings highlight a trade-off between segmentation accuracy and computational efficiency, underlining the importance of selecting architectures based on application-specific priorities: high accuracy for offline analysis versus lightweight models for real-time UAV monitoring.

Several aspects can be identified as limitations of the conducted study. First, the set of analyzed architectures could be expanded, or further research could be carried out to improve the performance of the currently evaluated models.

Second, the size of the training, validation, and test datasets could be increased. Railway infrastructure, particularly the surrounding vegetation, is highly complex, with a vast diversity of plant cover types and railway embankment structures. Expanding these datasets and annotating additional images would likely enhance the models' generalization capabilities, thereby improving the overall quality of the developed solution.

Third, a more powerful computational unit than the NVIDIA Jetson AGX Orin could be employed. Developing and utilizing a more efficient processing platform could lead to faster input image processing and result generation.

Fourth, the proposed approach could be transferred to other domains and applications. The models can be trained to monitor other types of linear infrastructure, such as roads, power lines, rivers, or shorelines. With the current solution already developed, it is feasible to adapt it to new use cases, provided that appropriate adjustments are made.

Fifth, reusing the validation set as the test set does not preserve the integrity of the evaluation, as a dedicated test set is required for an unbiased performance estimate. This methodological limitation should be considered when interpreting the reported results.

Sixth, the study did not include a quantitative analysis of performance metrics (e.g., FPS, latency, power consumption) for embedded deployment. Therefore, the contribution is limited to demonstrating deployment feasibility rather than providing a full performance evaluation or confirming real-time operation. This aspect may represent an important direction to consider in future research aimed at strengthening the practical relevance of the proposed approach.

Under the current requirements, the developed solution fully met the project objectives and paved the way for broader research in this area. The conducted study constitutes a solid foundation for further work, both within the current scope and in related fields where automated monitoring of linear infrastructure is required.

Acknowledgement

This research was supported by the Łukasiewicz Research Network – Institute of Aviation in the project entitled *Assessment and analysis of the potential use of remote object detection methods and condition identification of critical infrastructure* carried out by the Remote Sensing Department. The author expresses her gratitude for the support provided.

References

[1] P. Aela, H.-L. Chi, A. Fares, T. Zayed, and M. Kim. UAV-based studies in railway infrastructure monitoring. *Automation in Construction* 167:105714, 2024. doi:10.1016/j.autcon.2024.105714.

[2] P. Anilkumar, P. Venugopal, K. Lokesh, G. NagaJyothi, and M. Nanda kumar. AA-TransDeepLabv3+: a novel semantic segmentation framework for aerial images using adaptive and attentive based Transdeeplabv3+ with hybrid optimization technique. *Signal, Image and Video Processing* 19(225), 2025. doi:10.1007/s11760-024-03617-z.

[3] A. Chandramouli, H. Song, M. Liu, a. Damai, H. S. Narman, et al. Deep learning approaches for railroad infrastructure monitoring: Comparing YOLO and vision transformers for defect detection. In: *2025 IEEE 16th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 0205–0211, 2025. doi:10.1109/UEMCON67449.2025.11267623.

[4] A. Chaurasia and E. Culurciello. LinkNet: Exploiting encoder representations for efficient semantic segmentation. In: *2017 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, 2017. doi:10.1109/VCIP.2017.8305148.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40(4):834–848, 2018. doi:10.1109/TPAMI.2017.2699184.

[6] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. Rethinking atrous convolution for semantic image segmentation. arXiv, arXiv:1706.05587, 2017. doi:10.48550/arXiv.1706.05587.

[7] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 833–851, 2018. doi:10.1007/978-3-030-01234-2_49.

[8] C. Chenglin, W. Fei, Y. Min, Q. Yong, and B. Yun. Edge-enabled real-time railway track segmentation. arXiv, arXiv:2401.11492, 2024. doi:10.48550/arXiv.2401.11492.

[9] S. Chilamkurthy. segmentation_models.pytorch: Segmentation models. Python library with neural networks for image segmentation based on PyTorch. GitHub, 2019. `https://github.com/chsasank/segmentation_models.pytorch`. [Accessed: 2024].

[10] M. Di Summa, M. E. Griseta, N. Mosca, C. Patruno, M. Nitti, et al. A review on deep learning techniques for railway infrastructure monitoring. *IEEE Access* 11:114638–114661, 2023. doi:10.1109/ACCESS.2023.3309814.

[11] M. Giunta, V. Barrile, G. Leonardi, and E. Genovese. Comprehensive railway track monitoring using unmanned aerial systems (UASs) and building information modelling (BIM). In: *Computational Science and Its Applications – ICCSA 2025 Workshops*, vol. 15894 of *Lecture Notes in Computer Science*, pp. 407–419. Springer, 2025. doi:10.1007/978-3-031-97648-3_27.

[12] A. Kirillov, R. Girshick, K. He, and P. Dollár. Panoptic feature pyramid networks. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6392–6401, 2019. doi:10.1109/CVPR.2019.00656.

[13] Y. Kwon, W. Kim, and H. Kim. HARD: Hardware-aware lightweight real-time semantic segmentation model deployable from edge to GPU. In: *Computer Vision – ACCV 2024*, Lecture Notes in Computer Science, pp. 252–269, 2024. doi:10.1007/978-981-96-0963-5_15.

[14] H. Li, P. Xiong, J. An, and L. Wang. Pyramid attention network for semantic segmentation. arXiv, arXiv:1805.10180, 2018. doi:10.48550/arXiv.1805.10180.

[15] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, et al. Feature pyramid networks for object detection. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 936–944, 2017. doi:10.1109/CVPR.2017.106.

[16] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia. Path aggregation network for instance segmentation. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8759–8768, 2018. doi:10.1109/CVPR.2018.00913.

[17] M. Lopez-Montiel, D. A. Lopez, and O. Montiel. JetSeg: Efficient real-time semantic segmentation model for low-power GPU-embedded systems. arXiv, arXiv:2305.11419, 2023. doi:10.48550/arXiv.2305.11419.

[18] Y.-H. Na and D.-K. Kim. Deep learning strategy for UAV-based multi-class damage detection on railway bridges using U-Net with different loss functions. *Applied Sciences* 15(15):8719, 2025. doi:10.3390/app15158719.

[19] C. R. Nagarathna. Intelligent aerial surveillance for safer railways using machine learning. *International Journal of Innovative Research and Scientific Studies* 8(5):1160–1166, 2025. doi:10.53894/ijirss.v8i5.9077.

[20] S. Qiao, L.-C. Chen, and A. Yuille. DetectoRS: Detecting objects with recursive feature pyramid and switchable atrous convolution. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10208–10219, 2021. doi:10.1109/CVPR46437.2021.01008.

[21] X. Qin, Z. Zhang, C. Huang, M. Dehghan, O. R. Zaiane, et al. U2-Net: Going deeper with nested U-structure for salient object detection. *Pattern Recognition* 106:107404, 2020. doi:10.1016/j.patcog.2020.107404.

[22] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241, 2015. doi:10.1007/978-3-319-24574-4_28.

[23] C. Shen, J. Zhang, Y. Ji, T. Xu, L. Jiang, et al. Real-time semantic segmentation for UAV perspectives on embedded platforms. In: *Advanced Intelligent Computing Technology and Applications. ICIC 2025*, vol. 15842 of *Lecture Notes in Computer Science*, pp. 425–434. Springer, 2025. doi:10.1007/978-981-96-9863-9_36.

[24] K. Stypułkowski, P. Gołda, K. Lewczuk, and J. Tomaszewska. Monitoring system for railway infrastructure elements based on thermal imaging analysis. *Sensors* 21(11):3819, 2021. doi:10.3390/s21113819.

[25] P. Wang. x-unet: Implementation of a U-net complete with efficient attention as well as the latest research findings. GitHub, 2024. `https://github.com/lucidrains/x-unet`. [Accessed: 2024].

[26] L. Wen, Y. Peng, M. Lin, N. Gan, and R. Tan. Multi-modal contrastive learning for LiDAR point cloud rail-obstacle detection in complex weather. *Electronics* 13(1):220, 2024. doi:10.3390/electronics13010220.

[27] Y. Weng, Z. Li, X. Chen, J. He, F. Liu, et al. A railway track extraction method based on improved DeepLabV3+. *Electronics* 12(16):3500, 2023. doi:10.3390/electronics12163500.

[28] Y. Weng, J. Yang, C. Zhang, J. He, C. Peng, et al. An improved DeepLabv3+ railway track extraction algorithm based on densely connected and attention mechanisms. *Scientific Reports* 15:2556, 2025. doi:10.1038/s41598-024-84937-5.

[29] M. Xu, Y. Guo, and J. Luo. Lightweight feature pyramid networks for real-time semantic segmentation on edge devices. *IEEE Access* 10:33645–33655, 2022. doi:10.1109/ACCESS.2022.3161230.

[30] Z. Zhang and G. Li. UAV imagery real-time semantic segmentation with global–local information attention. *Sensors* 25(6):1786, 2025. doi:10.3390/s25061786.

[31] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang. UNet++: A nested U-Net architecture for medical image segmentation. In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support (DLMIA 2018)*, pp. 3–11, 2018. doi:10.1007/978-3-030-00889-5_1.