STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **LXVI**, Number 1, 2022
DOI: 10.24193/subbi.2022.1.05

A DYNAMIC APPROACH FOR RAILWAY SEMANTIC SEGMENTATION

ANDREI-ROBERT ALEXANDRESCU AND ALEXANDRU MANOLE

Abstract. Railway semantic segmentation is the task of highlighting rail blades in images taken from the ego-view of the train. Solving this task allows for further image processing on the rails, which can be used for more complex problems such as switch or fault detection. In this paper we approach the railway semantic segmentation using two deep architectures from the U-Net family, U-Net and ResUNet++, using the most comprehensive dataset available at the time of writing from the railway scene, namely RailSem19. We also investigate the effects of image augmentations and different training dataset sizes, as well as the performance of the models on dark images. We have compared our solution to other approaches and obtained competitive results with larger scores.

1. Introduction

Railway transportation is one of the most efficient modes of moving people and goods from one location to another [7]. The original train routes, which consisted of a small number of stops connecting one point of interest to another, were employed for industrial purposes. More stations were created to assist railway transit as more enterprises saw it as a viable way of carrying freight and passengers. As a result, there was a greater demand for routes between stations.

While numerous advances in scene understanding for autonomous driving have been made in recent decades, one subject has received little attention: *autonomous trains*. Such systems should require as little human intervention as possible. Although fully-autonomous metro systems exist in some modern cities, smart systems for long distance cargo trains are still to be developed.

Received by the editors: 16 July 2022.

2010 *Mathematics Subject Classification.* 68T10, 68T45.

1998 *CR Categories and Descriptors.* I.4.8 [**Image Processing and Computer Vision**]: Scene Analysis – *Object recognition*; I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding – *Intensity, color, photometry, and thresholding.*

Key words and phrases. Binary Semantic Segmentation, Encoder-decoder, Railway blades, Deep Learning.

At first glance, the task of building smart trains appears to be simpler to solve than the problem of smart cars or trucks. Trains have a limited range of motion due to the rails on which they travel, thus most of the autonomy consists of adjusting the speed based on different factors such as rail topology (curves, switches), obstacles or adverse weather conditions. In reality, it may be as difficult to solve the task of railway scene understanding as it is to solve the task for road scene understanding, since there are many different traffic signs and lights located in various places along the rail track.

When building a fully autonomous train, the semantic segmentation of the rails is an important aspect that must be considered. This task can be considered as a subproblem for more complex tasks such as detection of switches [12] or anomalies [11], adapting the speed of movement based on the topology of the rails or smart breaking in case of obstacles. It is critical to build a model that can accurately highlight the rails in an image with as few incorrect pixels as possible. This solution might be used in a safety-critical system where even the tiniest mistake could result in derailment or even crashes.

Currently, the task of rails detection can be solved by using two different approaches [22]. The first one implies using image processing techniques such as image edge detection to search for rail features. The second one consists of using deep convolutional neural networks with powerful semantic segmentation potential. This approach can extract edges, colors or textures of rails in more complex images with multiple rail intersections.

In this paper we propose an intelligent solution to the rails semantic segmentation problem using deep neural networks, which leads to better results when compared to the current literature on this problem using the most comprehensive dataset from the rail scene available. Our solution receives as input an image taken from the egocentric point of view of the train containing one or multiple rail tracks. The output is an image of the same size as the input containing white pixels for the rail blades and black pixels for everything else.

The aim of this paper is to answer the following research questions:

- How reliable are the proposed methods for semantic segmentation given a dynamic environment (i.e. the camera on the train)?
- How can we surpass the current state-of-the-art for the rails segmentation problem?

The structure of the paper is the following: Section 2 describes related methods used for solving this task, Section 3 presents the proposed approach for the rails semantic segmentation task and Section 4 describes the experiments and the obtained results. Section 5 concludes the paper by offering an overview of the work and some future considerations.

2. Related Work

The task of rail semantic segmentation has received some attention in the past years.

Wang et al. [24] developed an end-to-end model that combines feature extraction using the ResNet-50 backbone [8], followed by a fully convolutional network for detecting the railroad based on a custom Railroad Segmentation Dataset (RSDS) consisting of 3000 images of size 1920x1080 divided as it follows: 2500 images for training, 200 for validation and 300 for test. They achieve a reasonable inference time of 20FPS (Frames Per Second) by extracting both the rails and their interior area, namely the sleeper. The weights used by the ResNet50 backbone were trained on the ImageNet dataset [18] by performing a fine-tuning process. They obtain a mean IoU of 0.898 and a Dice score (F1 measure) of 0.868.

Zhen Tao et al. [22] use a deep neural network called *RailNet* for extracting the rail lines features from an image. This network is trained to generate the binary segmentation map of the rails, which is then processed together with the orginal image using a line fitting algorithm based on a sliding window technique (two-stage detector). Since the system will be used in real-time situations, the inference time represents a key factor to be taken in consideration. In order to obtain a fast inference time of 74FPS and allow the model to be used on memory-constrained devices, they use *Depthwise Convolutions*, which were initially introduced in [19]. They create a custom dataset called *RAWRail* containing 3000 railroad tracks pictures of size 640x360 in which there may be three different types of tracks: straight, curved to the left or curved to the right, 1000 samples for each type. The images are grouped together with the segmentation mask containing the two parallel rails as the positive class and the background as the negative class. They divided the dataset into training, validation and test sets following the ratio 0.9:0.05:0.05 and manage to obtain an accuracy of 98.6% on the test set.

Zendel et al. [26] have created the largest dataset presently available comprising annotated photos obtained from the egocentric perspective of trains, called RailSem19. They have applied deep learning approaches to solve the semantic segmentation task, employing the FRRN (Full-Resolution Residual Network) architecture, which was pre-trained on the Cityscapes dataset [4] and fine-tuned using 4000 training photos randomly selected from the dataset. Based on the ResNet50 backbone, this architecture comprises of an end-to-end model that combines feature extraction and semantic segmentation. The FRRN architecture comes in two flavors: FRRN A and FRRN B, which differ in terms of the input image size: FRRN A processes images of 256×512 pixels and FRRN B processes images of 512×1024 pixels. They have used the FRRN

B version with a 512×512 input size because it has a larger receptive field, performing better than FRRN A. For the semantic segmentation task, after 60 epochs of training, Zendel et al. have obtained an intersection over union (IoU) of 71.5% for the rails class. The images were resized to 512×512 and they used a batch size of 2 on a single RTX2080Ti GPU.

Li et al. [15] have also used the RailSem19 dataset for the semantic segmentation task, but on another architecture named *RailNet*. The authors have used a VGG-16 backbone [20] on top of which they have added an Information Aggregation Module (IAM) that builds a relationship between each row and column of pixels from the image to semantically segment the rail blades. The weights of this module are acquired in two ways: by using simple learnable weights (RailNet-LW) that get updated with the gradient descent process or by using attention-based weights (RailNet-AW) that work better with the unbalanced class distribution. They divided the RailSem19 dataset into 5000 images for training and 3500 for validation and managed to obtain a mean IoU of 0.54 and a mean recall of 0.89 on the validation set using the attention-based version. They have resized the images to 160x320 and have used an extended version of the focal loss [1].

Jahan et al. [11] have also used the RailSem19 dataset to perform semantic segmentation on the rails using deep learning architectures. They used a U-Net type architecture in which the feature extractor backbone was changed to either VGG [20] or ResNet [8] in order to increase its performance. The weights of these networks were pre-trained on the ImageNet dataset [18]. They also experimented with two types of loss functions: Weighted Binary Cross-Entropy and Focal Loss. Instead of working with grayscale images, Jahan et al. [11] used RGB images of size 892×596 with a batch size of 4 on two NVIDIA GeForce GTX 1080 Ti GPUs. They made use of image augmentation techniques to increase the performance of the models by using horizontal flipping, random noise, random brightness and random contrast. They obtain the best mean intersection over union (mIoU) of 52.78% after 46 epochs when training on 8390 images, validating on 60 samples and testing on 50 images.

3. Our Approach

In this section we present our approach for the rails semantic segmentation problem using U-Net architectures. We describe the formalization of the task, the chosen architectures, the dataset and the loss functions used for experimentation.

3.1. **Formalism.** We define a two-dimensional image as a bidimensional matrix with r rows and c columns with topology $I = \{1, ..., r\} \times \{1, ..., c\}$. Define $img : I \to O$, where O has one of the following forms based on the image type:

- RGB image: $O = \{0, ..., 255\}^3$;
- grayscale image: $O = \{0, ..., 255\}$;
- binary image: $O = \{0, 1\}$.

Therefore, $img(i, j) = o$, $o \in O$, $i \in \{1, ..., r\}$, $j \in \{1, ..., c\}$, where (i, j) is a pair of integers denoting the coordinates of the image and o represents its value at that position.

We define the segmentation mask or the ground truth as $Y_{GT} : I \rightarrow \{0, ..., K - 1\}$ where K represents the number of types of objects considered for segmentation or the number of different segments considered. Therefore, $Y_{GT}(i, j) = k, k \in \{0, ..., K - 1\}$. Similarly, the prediction given by the segmentation model can be defined as $Y_p : I \rightarrow \{0, ..., K - 1\}, Y_p(i, j) = k, k \in \{0, ..., K - 1\}$.

The semantic segmentation model considered in this paper can be formalized as an algorithm that takes as input an image img and outputs an image $mask$ where $img, mask : I \rightarrow O$.

3.2. **Architectures.** For the rails segmentation problem we have chosen models from the U-Net family, which feature an encoder-decoder architecture with skip connections between distanced layers. Although these types of architectures were designed to solve the semantic segmentation task for medical images, many studies have shown how well they work for other tasks as well [2, 14]. In our experiments we have considered two model architectures: U-Net [17] and ResUNet++ [13].

3.2.1. *U-Net.* This architecture has a U-like structure formed of a contracting path and an expansive path [17]. The contracting path, otherwise known as the downward or encoder path, is used to learn what features are present in the image, while the expansive path, known as the decoder path, is used to distinguish where the learnt features are located in the image. Between the two parts of the network, skip-connections are used in order to concatenate depthwise information from the downward path to the expansive path. In 2015, the ISBI cell tracking challenge[1] was won by the U-Net architecture, showing state-of-the-art performance at that time.

3.2.2. *ResUNet++.* This architecture, introduced in 2019 builds on top of U-Net [13]. It adds the following: Squeeze and Excite blocks, Atrous Spatial Pyramidal Pooling (ASPP), and attention blocks. The Squeeze and Excite blocks [9] are used to recalibrate channel-wise feature responses by explicitly modelling interdependencies between channels. Atrous Spatial Pyramidal Pooling [3] is used to capture contextual information at various scales. ASPP

[1]https://biomedicalimaging.org/2015/program/isbi-challenges/.

acts as a bridge between the encoder and the decoder part of the architecture. Attention Units are used to enhance the weights of some layers by learning on what parts of the image to focus more, i.e. to pay more attention to.

All of the mentioned architectures work with squared input images for which we choose 512×512 as size representation.

3.3. **Dataset.** RailSem19 [26] is the most extensive dataset from the railway scene at the time of writing, and we have used it in our study to train models that solve the rail blade semantic segmentation challenge. It consists of 8500 photos captured from the train's ego-view, having the size of 1920×1080 pixels. The images were captured in a variety of weather, lighting, and seasons in 38 different countries. Ground-truth masks for the rails segmentation procedure and bounding boxes for various elements from the railway scene are included in the samples. The dataset is imbalanced from the perspective of rails:background pixels ratio. For each pixel annotated as rail, there are ≈ 37 pixels annotated as background.

3.4. **Loss Functions.** We have tried to use different loss functions during the training step of our approach: *Binary Cross-Entropy*, *Weighted Binary Cross-Entropy*, *Tversky similarity index* and *Focal Tversky Loss function*. In the following we present the definitions of these functions.

Let y be the ground-truth and $\hat{y}$ the value predicted by the model. For the segmentation problem with two classes, *rails* and *background*, the Binary Cross-Entropy [6] loss function is defined as:

$$BCE(y, \hat{y}) = -(ylog(\hat{y}) + (1 - y)log(1 - \hat{y})). \tag{1}$$

Since the RailSem19 dataset used in our study is imbalanced, the Weighted Binary Cross-Entropy version of the loss was also considered:

$$WBCE(y, \hat{y}) = -(w * ylog(\hat{y}) + (1 - y)log(1 - \hat{y})), \tag{2}$$

where w is the weight represented by the ratio between background and rails pixels. This way, the loss gives more weight to the positive class (i.e., rails class in our study) when y=1, thus leading to a higher value of the loss function in cases the predicted value $\hat{y}$ is off. This allows the optimizer to improve the model predictions for the positive class.

The Tversky similarity index [23] was used to balance False Positives and False Negatives. This index is a generalization of 4.1, expressed as:

$$TI_c = \frac{\sum_{i=1}^{N} p_{ic} g_{ic} + \epsilon}{\sum_{i=1}^{N} p_{ic} g_{ic} + \alpha \sum_{i=1}^{N} p_{i\bar{c}} g_{ic} + \beta \sum_{i=1}^{N} p_{ic} g_{i\bar{c}} + \epsilon}, \tag{3}$$

where p_{ic} is the probability that pixel i belongs to class c and $p_{i\overline{c}}$ is the probability that pixel i does not belong to class c. Conversely, the same is true for g_{ic} and $g_{i\overline{c}}$ respectively.

Since small regions of interest (ROIs) do not contribute to the loss significantly, i.e. the value of the loss is smaller for such ROIs, the Focal Tversky Loss function (FTL) was poposed in the work of Abraham and Khan [1]. It is parametrized by γ to switch between easy background and hard ROI training examples. The Focal Tversky Loss function is defined as:

$$FTL_c = \sum_c (1 - TI_c)^{1/\gamma}, \tag{4}$$

where γ varies in the range $[1, 3]$. Abraham and Khan [1] hypothesize that using a higher α in the generalized loss function from Equation 4 improves the model convergence by shifting the focus to minimize False Negative predictions. They also mention the values they used: $\alpha = 0.7$ and $\beta = 0.3$.

3.5. **Our Semantic Segmentation Process.** The goal of our study is to solve a real problem, automatic rails identification in images, thus the data on which the chosen models will be used may come in different shapes and forms. The associated software is supposed to operate on images retrieved by a video camera placed on top of the train, aimed towards the upcoming rail track portion. It can be assumed that the format of the images is 16:9, however it is less likely that all of the feed of the cameras will be of the same resolution. Thus, we must ensure that the proposed method can be adapted to different resolutions.

In order to address this issue without changing the shape of the rails upon resizing the images to the appropriate sizes that can be fed into the network, an offline pre-processing step is performed on the semantic segmentation data.

As the original size of the images from the RailSem19 dataset is 1920×1080 and the chosen models work with squared images, the proposed crop area is a 1080×1080 square in the center of the image. An example of such a cropping can be seen in Figure 1.

This offline step allows for using images of different sizes for the inference time, which are resized to size 1080×1080 without changing the aspect ratio of the objects of interest (i.e. rails). The downside is that part of the image is left out, however the most important part containing the rails on which the train is moving on is preserved.

The images are then resized to 512×512 in order to be given as input to the models to be trained. The output will be an image of size 512x512 with black and white pixels where white pixels represent the rails and black pixels denote the background.

FIGURE 1. Example of cropping an image of size 1920×1080. The red square contains the cropped 1080×1080 area.

The ratio between background and rail annotated pixels for the cropped dataset is *36.09 : 1*, which is similar to the one obtained for the normal dataset that was *37 : 1*. Knowing this, the weight for the loss function will not be changed since the difference between 36.09 and 37 is not large.

4. RESULTS

In this section we present the experiments performed during our study for the rails semantic segmentation problem in order to answer the research questions. First, we describe the metrics used for the evaluation. Afterwards we present the overfitting procedure performed to check the correctness of the selected architectures. Lastly, we describe the settings for each experiment and the obtained results.

4.1. **Metrics.**

In order to evaluate the obtained results, the following metrics were used:

- *Intersection over Union (IoU)* also known as the Jaccard metric [10], is the most used evaluation metric in object segmentation. It is used to determine True Positives and False Positives in a given set of predictions. True positives (TP) represent those data samples that were predicted correctly by the classifier to be a positive class, while false positives (FP) represent the samples that were incorrectly labeled as positive by the model.

Considering an input image X, its corresponding ground-truth mask Y, and the model M_{seg}, we can obtain Y_p as $M_{seg}(\mathrm{X}) = Y_p$. In order to define the mean IoU, denoted as $mIoU$, Y and Y_p are used as follows:

$$mIoU = \frac{|Y \cap Y_p|}{|Y \cup Y_p|}. \tag{5}$$

This formula computes the mean IoU by considering all K types of objects to be segmented. In order to compute the IoU for a single class, the following formalism can be applied. Let k, $k \in \{0, 1, ..., K-1\}$ be the class for which we wish to compute the IoU score, then:

$$IoU_k = \frac{|\{(i,j)|i,j \in \mathbb{N}^*, Y(i,j) = Y_p(i,j) = k\}|}{|Y \cup Y_p|}, \tag{6}$$

where the tuple (i, j) can be interpreted as an (x, y) coordinate in a two-dimensional image.

We have defined two metrics, one for each class: the *IoU Rail* for the rails and *IoU Background* for the background. All of the *IoU* metrics range from 0 to 1, where 0 means no intersection and 1 means perfect overlap.

- Dice (F1) Score is based on the Dice coefficient, which was first introduced in [5]. This metric is used to measure the overlap between two samples and is equivalent to the F1 score in a binary context. The metric ranges from 0 to 1 where 1 denotes the absolute complete overlap. Using the previously defined notations, the Dice Coefficient can be expressed as in Equation 7:

$$DC = 2\frac{|Y \cap Y_p|}{|Y| + |Y_p|}. \tag{7}$$

4.2. **Experiments.** In order to check the appropriateness of the chosen architectures for the rails semantic segmentation problem we have performed different experiments. We wanted to compare the results obtained by our models with the ones available in the literature (experiments A and B) which use the RailSem19 corpus on similar data distributions. We have also tested the performance of the U-Net model on dark images and tried to enhance the performance by adding random brightness augmentations (experiment C).

The tables with results follow a similar column structure: **Model** denotes the model used for experimentation, **Parameters** denotes the number of learnable parameters corresponding to each model, **mIoU** represents the mean IoU score, **IoU Rail** and **IoU Bg** denote the IoU scores for the rails and background classes, and column **Dice** contains the Dice scores.

Experiment A. In the first experiment, we have compared the results obtained by our models with the ones obtained by Zendel et al. [26] in their study. Zendel et al. used a data distribution of 4000 samples, 3000 for training, 500 for validation, and 500 for testing. We did not approach the problem using this three-way split. However we have split 3500 samples selected randomly into 3000 samples for training and 500 samples for validation, and we report the results on the validation dataset. The results obtained using U-Net and ResUNet++ are showcased in Table 1.

Column **Model** contains the model trained and validated. Our models have either the w1 or w5 suffix, denoting the used weight value for the Weighted Binary Cross-Entropy loss.

TABLE 1. Rail semantic segmentation results with a 3000:500 random data distribution.

Model	**Parameters**	**mIoU**	**IoU Rail**	**IoU Bg**	**Dice**
U-Net w1	30 M	**0.81**	0.63	0.98	0.77
U-Net w5	30 M	0.80	0.61	0.98	0.76
ResUNet++ w1	**14** M	0.80	0.61	0.98	0.76
ResUNet++ w5	**14** M	0.78	0.59	0.98	0.74
FRRNB [26]	16 M	0.71	-	-	-

In Table 1, it may be observed that the results obtained did manage to surpass the results obtained by Zendel et al., which are included in Table 1 in the last row. The architectures considered in our study function in a similar way to FRRNs [16] by exploiting residual connections for helping localization of pixels. Despite this, U-Nets lead to better results. Moreover, we have obtained these results without pre-training the networks on CityScapes [4]. Both U-Net and ResUNet++ surpass Zendel et al.'s results, with ResUNet++ having the least number of parameters. The U-Net model obtains slightly better results, however it utilizes approximately double the number of parameters. A size-performance trade-off must be made between the two.

For this experiment, the images size was 512×512, similar to the ones used in Zendel et al's study. The weight decay was set to $1e^{-3}$ and dropout layers were used with probability 0.2. Although counter-intuitive, the unweighted loss function with a weight of 1 leads to slightly better results in less time. An explanation for this might be that using a larger weight for the rails class, the loss is penalised harder, thus decreasing the learning speed. The models were trained for 32, 36, 22, 38 epochs respectively in the order presented in Table 1. These values were chosen after training until reaching a plateau in the loss value, meaning that no further improvements could be obtained.

The ResUNet++ architecture takes longer to reach the performance of U-Net because it has more complex components.

Experiment B. In the second experiment, we have compared our selected models to the RailNet architecture from Haoran Li et al. [15]. For this, 5000 images were randomly sampled for training, while the remaining 3500 were used for validation. The outcome of this experiment is given in Table 2.

TABLE 2. Rail semantic segmentation results on a 5000:3500 random data distribution.

Model	Parameters	mIoU	IoU Rail	IoU Bg	Dice
U-Net w1	30 M	**0.81**	0.64	0.98	0.78
U-Net w5	30 M	0.70	0.44	0.97	0.61
ResUNet++ w1	**14** M	0.79	0.60	0.98	0.75
ResUNet++ w5	**14** M	0.78	0.58	0.98	0.74
RailNet [15]	138 M	0.54	-	-	-

Surprisingly, the results obtained using the 5000:3500 data distribution are considerably larger than the ones obtained by Haoran Li et al. [15]. These results were obtained using the same random distribution of samples. All models were trained with a batch size of 4 for 20 to 40 epochs. Similar to previous experiments, the models trained with a loss function weight of 1 lead to better results. The best results were selected. Resulting samples are visible in Figures 2 and 3, for U-Net and ResUNet++ respectively.

The encoder-decoder architecture with skip connections that is represented by U-Net and ResUNet++ appears to be performing better on this task than the VGG architecture combined with an Information Aggregation Module used by Haoran Li et al. [15]. It also has fewer parameters.

Experiment C. The last experiment was performed in order to better understand how the model behaves in dark conditions. For the training set, both daylight images and dark images were considered. To be more precise, a total of 3500 images were selected consisting of 3095 day images and 405 dark images. The train:validation split was 80:20. The training set consists of 2467 day images and 619 dark images, while the validation set contains 324 day images and 81 dark images.

Three different validation sets were considered: the one previously detailed, one with day images exclusively and one with dark images only. The results of these experiments are given in Table 3, where DN denotes day and night samples, D denotes day only samples, and N denotes night only samples.

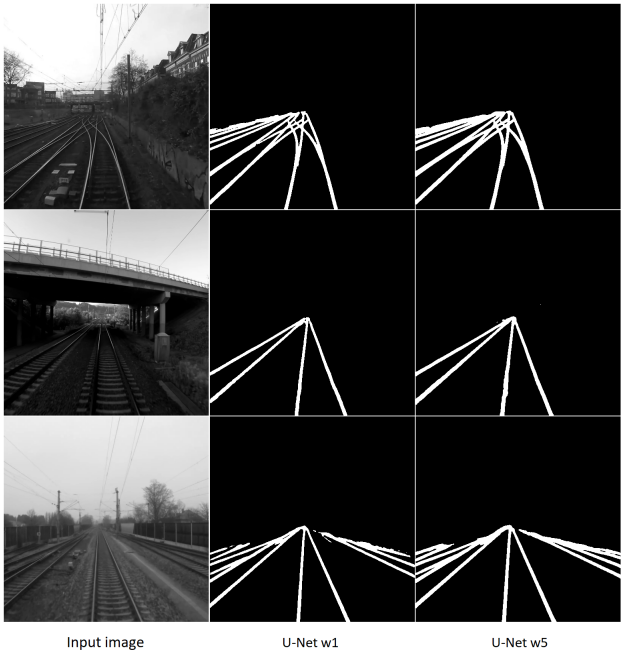

FIGURE 2. U-Net results on the 5000-3500 distribution.

TABLE 3. Results for the rails semantic segmentation task on dark images with and without augmentations.

Val. Distribution	Model	mIoU	Rail IoU	IoU Bg	Dice
DN	U-Net	0.77	0.57	0.97	0.72
DN	U-Net Aug	0.77	0.57	0.97	0.72
D	U-Net	0.77	0.57	0.97	0.73
D	U-Net Aug	0.77	0.57	0.97	0.73
N	U-Net	0.75	0.53	0.97	0.69
N	U-Net Aug	0.76	**0.54**	0.97	0.70

In order to address the issue of poor luminosity, one more training attempt was made with augmentations, which would randomly decrease the brightness of the grayscale images by decreasing from each pixel value a constant in order to make them darker. This constant was set to 80 after manually testing multiple values. This augmentation can be interpreted as transforming day images into night images.

As expected, the results are not as high as those for the previous experiments using approximately 3500 images in total. One of the reasons for this low

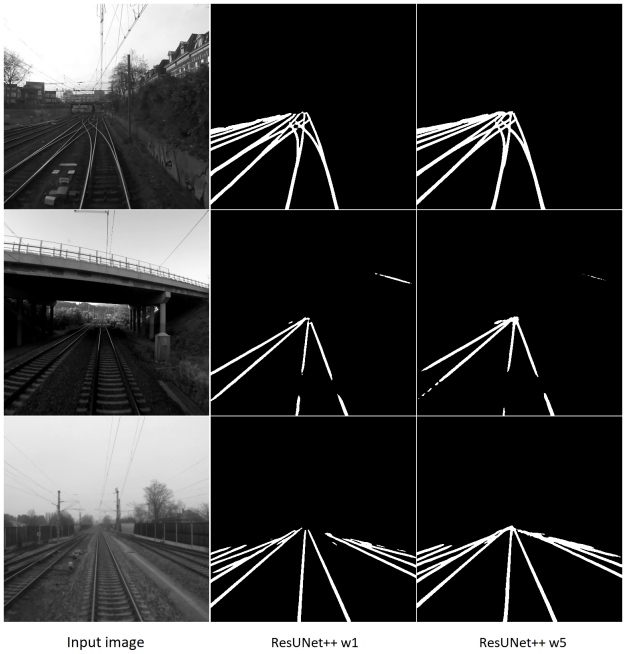

FIGURE 3. ResUNet++ results on the 5000-3500 distribution.

performance is due to the nature of the task: it is more difficult to segment rails in poor luminosity conditions.

An increase of 1.3 percentages in the Rail IoU score is observed when performing the evaluation on night-only images using random brightness augmentations. This outcome was expected since this type of augmentation helps the model learn better representations for darker images.

4.3. **Analysis.** We have observed that the U-Net architecture leads to better results than ResUNet++, although the latter uses more advanced features designed to improve its performance. Nevertheless, both architectures are appropriate for obtaining usable results for the rail semantic segmentation task. We have also observed that a smaller weight term for the Binary Cross-Entropy loss function leads to better results in less training time. This was expected since a smaller weight means a lower penalisation when the background class is predicted for a pixel instead of the rails class.

Since the variation between the results obtained using U-Net and ResUNet++ is quite small, it is difficult to proclaim a definitive winner. The decisive factor one could consider when comparing two network architectures would be the size of the network. From this perspective, ResUNet++ is deemed to

be better, having fewer trainable parameters. In the end, both architectures accomplish the task of semantic segmentation on rails well enough.

To answer the first research question, we may assume that using the ResUNet++ architecture, which has less parameters (14M) than U-Net (30M), is suitable for a dynamic environment such as the camera on the train. To add to this, the dataset used for training contains images in multiple weather conditions (snow, rain, smog) and at different times of day and night, thus increasing the usability of the trained models.

Using U-Net like architectures is the answer to the second research question, since we obtained better numerical results than three other works from the literature that aim to solve the same task. On top of the original architecture implementations, we added Dropout layers [21] and considered weighted loss functions for optimizing the models.

5. Conclusions and Future Considerations

In this article, an efficient solution was presented for solving the rails semantic segmentation task using deep architectures from the *U-Net* family on images taken from the perspective of the train. The considered architectures, namely *U-Net* and *ResUNet++*, led to some competitive results when compared to a selected range of related works. Our results surpassed the state-of-the-art on this task by 9 percentages.

Despite this, the task is still challenging for fine-grained semantic segmentation of rails that are further away from the camera. Other issues that are still open for research include segmenting rails in dark places (night, tunnels) or avoiding False Positives such as shadows or other objects similar to rails.

In the future, multiple aspects can be improved:

- The dataset aggregated by Zendel et al. [26] contains some images that lack proper annotations for the rails class, meaning that some rails are not annotated correctly. An improvement would be to annotate these missing rail blades and to increase the number of samples.
- From the perspective of the considered architectures, more tests can be performed with even more semantic segmentation architectures in order to compare them and select the most appropriate one for the rails semantic segmentation problem. Maybe even try a novel architecture designed especially for this task.
- Since the images of the rails are provided by a camera placed on top of the train, there is a high similarity between each frame: the rail blades are almost in the same position, but slightly shifted between consecutive frames. For this reason, an architecture that considers the previous pixels classified as rails might perform better on this task. An example of such an architecture is introduced in [25].

- It would be useful to measure the FPS of the mentioned methods and analyze what would be the most suitable hardware to be used on a real train from the perspective of power consumption and feasibility.

References

[1] Nabila Abraham and Naimul Mefraz Khan. "A novel focal tversky loss function with improved attention u-net for lesion segmentation". In: *International Symposium on Biomedical Imaging*. IEEE. 2019, pp. 683–687.

[2] Rytis Augustauskas, Arūnas Lipnickas, and Tadas Surgailis. "Segmentation of Drilled Holes in Texture Wooden Furniture Panels Using Deep Neural Network". In: *Sensors* 21.11 (2021), p. 3633.

[3] Liang-Chieh Chen et al. "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs". In: *TPAMI* 40.4 (2017), pp. 834–848.

[4] Marius Cordts et al. "The cityscapes dataset for semantic urban scene understanding". In: *CVPR*. 2016, pp. 3213–3223.

[5] Lee R Dice. "Measures of the amount of ecologic association between species". In: *Ecology* 26.3 (1945), pp. 297–302.

[6] Irving John Good. "Rational decisions". In: *Breakthroughs in statistics*. Springer, 1992, pp. 365–377.

[7] Christian Growitsch and Heike Wetzel. "Testing for economies of scope in European railways: an efficiency analysis". In: *Journal of Transport Economics and Policy (JTEP)* 43.1 (2009), pp. 1–24.

[8] Kaiming He et al. "Deep residual learning for image recognition". In: *CVPR*. 2016, pp. 770–778.

[9] Jie Hu, Li Shen, and Gang Sun. "Squeeze-and-excitation networks". In: *CVPR*. 2018, pp. 7132–7141.

[10] Paul Jaccard. "The distribution of the flora in the alpine zone. 1". In: *New phytologist* 11.2 (1912), pp. 37–50.

[11] Kanwal Jahan, Jeethesh Pai Umesh, and Michael Roth. "Anomaly Detection on the Rail Lines Using Semantic Segmentation and Self-supervised Learning". In: *SSCI*. IEEE. 2021, pp. 1–7.

[12] Kanwal Jahan et al. "Deep Neural Networks for Railway Switch Detection and Classification Using Onboard Camera Images". In: *SSCI*. IEEE. 2021, pp. 01–07.

[13] Debesh Jha et al. "Resunet++: An advanced architecture for medical image segmentation". In: *2019 IEEE International Symposium on Multimedia (ISM)*. IEEE. 2019, pp. 225–2255.

[14] Martin Kolařík et al. “Optimized high resolution 3D dense-U-Net network for brain and spine segmentation”. In: *Applied Sciences* 9.3 (2019), p. 404.

[15] Haoran Li et al. “RailNet: An Information Aggregation Network for Rail Track Segmentation”. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2020, pp. 1–7.

[16] Tobias Pohlen et al. “Full-resolution residual networks for semantic segmentation in street scenes”. In: *CVPR*. 2017, pp. 4151–4160.

[17] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. “U-net: Convolutional networks for biomedical image segmentation”. In: *International Conference on Medical image computing and computer-assisted intervention*. Springer. 2015, pp. 234–241.

[18] Olga Russakovsky et al. “Imagenet large scale visual recognition challenge”. In: *International journal of computer vision* 115.3 (2015), pp. 211–252.

[19] L Sifre and Stéphane Mallat. “Rigid-motion scattering for texture classification”. In: *Phd. Thesis* (2014).

[20] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *International Conference on Learning Representations*. 2015.

[21] Nitish Srivastava et al. “Dropout: a simple way to prevent neural networks from overfitting”. In: *The journal of machine learning research* 15.1 (2014), pp. 1929–1958.

[22] Zhen Tao et al. “Accurate and Lightweight RailNet for Real-Time Rail Line Detection”. In: *Electronics* 10.16 (2021), p. 2038.

[23] Amos Tversky. “Features of similarity.” In: *Psychological review* 84.4 (1977), p. 327.

[24] Yin Wang et al. “RailNet: A segmentation network for railroad detection”. In: *IEEE Access* 7 (2019), pp. 143772–143779.

[25] Rui Yao et al. “Video object segmentation and tracking: A survey”. In: *ACM Transactions on Intelligent Systems and Technology (TIST)* 11.4 (2020), pp. 1–47.

[26] Oliver Zendel et al. “Railsem19: A dataset for semantic rail scene understanding”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 2019, pp. 32–40.

Department of Computer-Science, *Faculty of Mathematics and Computer Science*, *Babeș-Bolyai University*, Cluj-Napoca, Romania

Email address: andrei.alexandrescu@stud.ubbcluj.ro

Email address: alexandru.manole@stud.ubbcluj.ro