Towards Resource-Efficient Deep Learning for Train Scene Semantic Segmentation

Marie-Claire Iatrides[1,2], Petra Gomez-Krämer[1], Olfa Ben Ahmed[3] and Sylvain Marchand[1]
[1]*L3i Laboratory, La Rochelle University, La Rochelle, France*
[2]*Association Ferrocampus, Saintes, France*
[3]*Xlim Institute of Research, Poitiers University, Poitiers, France*
{*marie-claire.iatrides, petra.gomez, sylvain.marchand*}*@univ-lr.fr, olfa.ben.ahmed@univ-poitiers.fr*

Keywords: Lightweight CNN, Resource-Efficient ML, Semantic Segmentation, Deep Learning, Train Environment.

Abstract: In this paper, we present a promising application of scaling techniques for segmentation tasks in a railway environment context to highlight the advantages of task specific models tailored for on-board train use. Smaller convolutional neural networks (CNNs) do not focus on accuracy but resource efficiency. Our models are scaled using skip connections as well as quantization in order to form lightweight models trained specifically for our context. The proposed models have been evaluated both in terms of segmentation performance and efficiency on state of the art scene segmentation datasets namely RailSem19 and Cityscapes. We have obtained models with less than 3.5M parameters and a minimum of 78.4% of segmentation accuracy showing that lightweight models can effectively segment the railway surroundings.

1 INTRODUCTION

Autonomous systems are the next step towards optimizing transportation systems. The railway industry is one of the most developed transportation modes in France with over 27,483 km of rail tracks operated by the SNCF (Société Nationale des Chemins de Fer français). Because of its direct interactions with nature, the railroad network requires considerable maintenance. Another challenge is that a lot of the technology involved in our railroad system has not undergone major renovations and therefore presents numerous possibilities for innovation. For instance, some active regional express railways date back to the 1980s and little improvement has been made on parts of the infrastructure. In this context, there are several major challenges to tackle the development of both practical and environment friendly systems in the railway industry. One of the major concerns of railway operations is the maintenance of the vegetation in the rail surroundings. Vegetation has multiple impacts on the rail traffic: it creates obstacles on the rails, endangers ballast stability or can even cause derailments. As a result, it is one of the main concerns for smaller regional lines that are not isolated like most high speed rails and are in direct interaction with nature.

We present in this article a study of resource-efficient models for the detection and segmentation of the train's environment. We aim to develop an on-board system that uses solely images as input to elaborate an analysis of the train's surrounding. We focus on lightweight models to limit energy consumption, CO_2 emissions and optimize memory space on-board. With this in mind, semantic segmentation of the scene allows for the detection of the different elements of the environment. As an embedded system, it should follow a certain set of constraints, mainly regarding memory consumption, computer power and inference. We therefore explored scaling techniques in order to reduce the impact of our models. Scaling can be approached through multiple angles such as limited input information, model size and memory consumption. The aim is to determine whether images are sufficient to perform predictive maintenance on vegetation in the environment of trains with a frugal deep learning approach. To that end, the contributions of our work are a comparative study for resource-efficient convolutional neural networks (CNNs) for semantic segmentation, an adaptation of scaling techniques from classification tasks to segmentation, and then, a study of the effects of quantization on these CNNs and their limitations with more complex datasets.

The remainder of this article is structured as follows. Firstly, we explore previous research related to our task. Secondly, we define the perimeter we have

Iatrides, M.-C., Gomez-Krämer, P., Ben Ahmed, O. and Marchand, S.
Towards Resource-Efficient Deep Learning for Train Scene Semantic Segmentation.
DOI: 10.5220/0013139000003912

In *Proceedings of the 20th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2025) - Volume 3: VISAPP*, pages 347-354
ISBN: 978-989-758-728-3; ISSN: 2184-4321

chosen for our work and its application. And then, we describe our approach to introduce our results. Lastly, we discuss the implications of the said results in order to conclude on their implications towards future work.

2 RELATED WORK

In this section, we briefly review related work with respect to autonomous trains and the maintenance of their infrastructure. Furthermore, we discuss frugal techniques in machine learning applications and image segmentation methods.

2.1 Context

In recent years, there has been much progress in autonomous driving surveillance systems (Zakaria et al., 2022), but most are limited to either regular road vehicles or trains operating in controlled environments such as subways (Singh et al., 2021). While similar, operating conditions are much more complicated for outdoor trains and require more frequent maintenance. In a previous work (Skibicki and Licow, 2022), researchers working with the Swedish railway industry developed a vision-based method to detect weed presence on tracks to evaluate the infestation level. To the best of our knowledge, the research community has yet to produce a complete system to ensure maintenance of the surrounding vegetation in the railway environment. Many works focus solely on the application of autonomous systems for high speed rail (Yin et al., 2020). On the other hand, research on open environment lines are still very much lacking. There are multiple ways to implement autonomous systems in trains (Singh et al., 2021), each utilizing different sources of information and applying a wide variety of processing operations. A systematic review of open-source datasets for railroad applications (Pappaterra et al., 2021) clearly shows that, while many domains have been well explored for which data was made available, there is still a lack of data for image segmentation. In his thesis work (Duquene, 2023), Duquene explores the use of imitation and reinforcement learning architectures to create an autonomous driving system for trains, primarily for speed control.

2.2 Image Segmentation

Previous works, such as RailNet (Wang et al., 2019), introduced a deep-learning-based model for the binary semantic segmentation of railway environments designed for railroad appreciation. This model shows a high detection rate, but a relatively low accuracy. The introduction of the SqueezeNet (Iandola et al., 2016) architecture brought about the development of light and fast semantic segmentation networks like SqueezeUNet (Beheshti and Johnsson, 2020). This work introduces the fire module that both reduces the size of the model compared to a regular UNet and has a 17 times lower inference. Another comparable work is the Squeeze-SegNet (Nanfack et al., 2017) that fuses the enhancing methods brought by SegNet (Badrinarayanan et al., 2017) with those of the fire module.

2.3 Scaling of Resources

Following an introduction to frugality in machine learning (Evchenko et al., 2021), there are three main aspects of frugality in this domain: cost of data, computation process requirements and model characteristics. There can be multiple approaches to this objective, but the most evident is to reduce the size of the model as a whole, without modifying its structure. We call this method quantization. It converts stored model weights into a smaller format. In a previous work, authors achieve a 4 to 8 times reduction of a CNN's size through quantization and pruning (Ahamad et al., 2022) with minimal drop in accuracy. In another thesis work, the author explored the three types of quantization: data-free, gradient-based post-training quantization (GPTQ) and quantization-aware training (Yvinec, 2023). These methods show a significantly unstable trade-off between accuracy and speed. Another method to reduce the size of a model is through its architecture, for example, skip connections.

Before the introduction of skip connections, models were limited in their depth by the vanishing gradient. ResNet (He et al., 2016) was one of the first works to find a solution to this challenge with the introduction of the residual connection. EfficientNet (Tan and Le, 2019) took this idea even further through balance and optimization of smaller models for object classification. Another significant work is the UNet (Ronneberger et al., 2015) which implements long skip connections between the encoder and decoder blocks. In the field of classification, SqueezeNet (Iandola et al., 2016) also paved the way for small model architectures by introducing the fire module. And lastly, DenseNet (Huang et al., 2017) introduced another type of skip connections with the dense block. It allows for better performance with smaller models and makes better use of the existing parameters. In our work, we have modified such models to be used and compared in a semantic segmentation task. To do

so, we adapt the methods discussed in this section to a specific application, the railroad environment. This context brings strict requirements in model, input and memory size. As such, these models and techniques show clear potential for our application. Our approach also aims to determine the limit of model size reduction using the aforementioned techniques and to test the performances of our models under hard resource constraints.

3 PERIMETER OF STUDY

In railway tasks, there is still a lack of open-source datasets. Existing image datasets for scene understanding are mainly limited to the detection of signs, i.e. FRSign (Harb et al., 2020), of rails, i.e. RailSet (Zouaoui et al., 2022) or Rail-DB (Li and Peng, 2023), and thorough semantic segmentation, i.e. RailSem19 (Zendel et al., 2019). We describe in the following the task we address in this article as well as the dataset we used.

Segmentation can serve to detect rails, but also provides a gateway to infrastructure surveillance and predictive maintenance. Analyzing what surrounds the train allows for a more comprehensive system that can adapt to its surroundings. As such, this task can lead the system to prevent obstacle collision, infrastructure degradation or even critical failure of the railway system. RailSem19 (Zendel et al., 2019) is introduced as a complete dataset with 8,500 images containing dense pixel-wise annotations for 19 classes. Varying weather and lighting conditions are represented as well as speed associated blur effects and lighting artifacts due to tunnels and nighttime capture. Images are extracted from video sequences with an ego-vehicle point of view with a 1920×1080 pixel resolution.

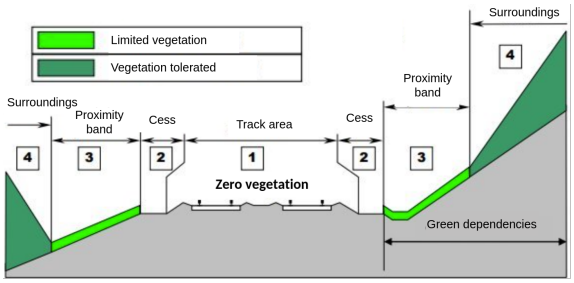

Figure 1: Segmentation of a railway section following SNCF regulations. (© SNCF Réseau, modified).

This dataset is inspired by the Cityscapes dataset both in terms of classes and labeling policy. Some images of trams and city railways are even taken from Cityscapes to form RailSem19. RailSem19 contains a thorough analysis of the elements comprised in the railroad environment with an accurate declination of its signalization system. All in all, these elements are essential in a global analysis of the environment of trains, but we are not interested in the elements that provide traffic information in our task. For the analysis considered in this work, the original 19 classes were modified to fit into 8 classes (Table 1). These classes represent the zones of interest defined by the SNCF for the analysis and evaluation of the vegetation in the railroad vicinity (Figure 1). Henceforth, the modified dataset will be referred to as RailSem8.

4 SEMANTIC APPROACH

This work originates from the need of an industrial application for predictive maintenance and environment perception in trains. In order to solve this task, we focus on the surveillance of vegetation and its impact on the region of circulation. In order to perceive the environment, this work implements semantic segmentation of images taken from the ego-perspective of trains. In this section, we present the different models that where selected following the energy, storage and computation power constraints necessary for onboard systems. We also explore the implementation of scaling techniques to the models for the development of resource-efficient deep learning techniques. Semantic segmentation provides a pixel-wise classification into a class through mapping. As a result, it provides a detailed description of the entities that constitute the environment of trains. While this task is relatively costly in terms of computation, results can provide fine-grained information on the input content.

4.1 Scaling Techniques

Scaling techniques were explored to limit the model's size and parameters without compromising results. This work mainly focuses on the use of skip connections to limit the degradation of gradients and conserve high details. The downside of this architecture is that the number of parameters is still relatively high to be considered resource-efficient. For example, a ResNet-18, the smallest publicly available network using this type of architecture, has a total of 11.4M parameters. As a result, this type of skip connection is not further explored in this work. Dense connection-based architectures like DenseNet (Huang et al., 2017) networks have a relatively low number of parameters with similar performances. For example, DenseNet-56, with 56 layers as opposed to the original DenseNet-121, only counts 1.5M parame-

Table 1: RailSem19 modified dense labels to RailSem8; **In Frames** = %age of frames with pixels in the corresponding class.

Label RS8	Rails	Track	Track-bed	Terrain
Label RS19	rail-raised, rail-embedded	tram-track, rail-track	track-bed	terrain
In Frames	100.0%	100.0%	87.6%	61.2%
Label RS8	Vegetation	Sky	Obstacles	Background
Label RS19	vegetation	sky	human, car, truck, on-rails	road, sidewalk, construction, fence, pole, traffic-light, traffic-sign, void
In Frames	83.3%	94.5%	13.8 $\sim$ 24.4%	72.1 $\sim$ 100.0%

ters. As a result, this technique seems very promising for the task at hand in terms of model scaling. The last method is the long skip. UNets are comparably smaller models by design and are optimized for fast processing, performance in image analysis and generalization purposes. In order to profit from these characteristics, this type of models was also explored in this work.

4.2 Models

Four networks were implemented for this work: UNet, Squeeze-UNet, DenseNet-67 and DenseNet-56. All of these models are built with 4 encoder blocks and decoder blocks except DenseNet-67, which has 5 encoder and decoder blocks as described in Table 2. Based on the UNet (Ronneberger et al., 2015) architecture an original version was implemented for this project. While being relatively big, UNet has a much faster inference time. The model was adapted from the aforementioned methods and modified to a UNet-like architecture in order to compare it to the other architectures explored in this work. We implemented our own Squeeze-UNet based on the description of the SqueezeSegNet, a modified SqueezeNet. On the other hand, DenseNet was originally developed for classification with 121 layers (Huang et al., 2017). In this work, we modified and implemented two versions, much smaller in scale, DenseNet-56 and DenseNet-67. These models have much fewer operations and parameters (Tables 3 and 4) and thus show promise in terms of memory efficiency. All selected models were chosen as archetypes of scaling techniques in terms of inference speed, performance improvements, and have shown very promising results in similar applications.

In our work, we pre-processed data with size reduction in order to test the limits of input resolution towards performance. We know that the size of process during training is directly proportional to the size of data. When reducing the image resolution, we inevitably reduce the model's capacity to recognize fine details which then impacts the results and performance. On the other hand, smaller images implies higher batching of images is possible during training. Increasing the batchsize allows for a better generalization during training which then improves performance. All in all, the preprocessing allows for smaller processes and improves generalization.

5 EXPERIMENTS AND RESULTS

All models in this work were trained from scratch on a NVIDIA RTX A-6000 GPU with a 2.45GHz processor and AMD EPYC 7763 CPUs installed with 1To of RAM. During initial tests, the models were trained for 100 epochs and then for 200 epochs. All training was done with a 1e-5 learning rate, a batchsize of 8 for training and 4 for validation. The loss function is a cross entropy loss minimized through an Adam optimizer. These parameters were chosen through empirical analysis in order to find the best combination of input parameters and evaluation processes. For our experiments, we divided the datasets into 60-20-20 proportions for training, validation and testing sets with a random split manual seed set at 0 for repetition. As for data pre-processing, we have worked to enforce input efficiency by reducing the resolution of the image from its original 1920×1080 to 320×240.

5.1 Model Comparison

As presented in Table 4, most models studied are much smaller in terms of size compared to the state-of-the-art shallow network, EfficientNet (Tan and Le, 2019) that counts 11.3M parameters. Squeeze-UNet, counting 8.1M parameters (Table 4), was found to perform poorly (Table 5) with a 0.691 Dice Score compared to a minimum of 0.78 for the others and returns masks that visually reveal a heavy error rate (Figure 4) and was therefore pushed aside from our analysis. Results in Figure 4 show there is a weakness towards night view images, but visually have a high performance even under rain or fog.

We also performed a convergence analysis of our models over 100 epochs (Figure 2). The model was tested at each modification of the weights after the validation step. From this graph, we can see a rapid increase of the quality of the performance according to the Dice score in the early stages of training and after the 20^{th} iteration, the system stabilizes on a linear convergence curve with $R^2 = 0.85$. Later results in Dice score, after 200 epochs of training (Table

Table 2: Model architectures through blocks, each model is implemented with 4 or 5 encoder and decoder blocks.

Blocks	UNet	Squeeze-UNet	DenseNet-56/-67
Encoder	[Conv3x3 + BatchNorm + ReLU] x2 MaxPool2x2	[FireLayer] x2 Conv4x4	[BatchNorm + ReLU + Conv3x3 + Dropout] x4 (x5) [BatchNorm + ReLU + Conv1x1 + Dropout + MaxPool]
BottleNeck	Conv1x1	[FireLayer] x2	[BatchNorm + ReLU + Conv3x3 + Dropout]
Decoder	ConvTranspose2x2 [Conv3x3 + BatchNorm + ReLU] x2	TransFireLayer [FireLayer] x2	ConvTranspose3x3 [BatchNorm + ReLU + Conv3x3 + Dropout] x4 (x5)

Table 3: Model size and operation counts.

Model	UNet	Squeeze-UNet	DenseNet-56/-67	
Params	31.0 M	8.1 M	1.4 M	3.5 M
FLOPs	$118.2\text{x}10^9$	$111.5\text{x}10^9$	$27.6\text{x}10^9$	$64.5\text{x}10^9$
MACs	$59.0\text{x}10^9$	$55.6\text{x}10^9$	$13.6\text{x}10^9$	$31.9\text{x}10^9$

Table 4: Model characteristics.

Model	Inference time(ms)	Check-point file(MB)	Process(GB)	Parameters
UNet	5.5	124.3	**1.49**	31.0 M
Squeeze-UNet	10.0	32.6	1.84	8.1 M
DenseNet-67	17.8	14.2	9.50	3.5 M
DenseNet-56	12.2	**6.2**	3.17	**1.4 M**

Figure 2: DenseNet-67 convergence graph over 100 epochs.

4), suggest a diminution of the learning curve with a reduction of the impact of additional epochs on the model's performance. Between 100 and 200 epochs for DenseNet-67, the Dice score increases of only 0.042 which gives a linear coefficient of $R^2 = 0.42$ after 100 epochs thus further supporting this conclusion. It also shows a risk of over-fitting with further training for this model.

Table 5: Model's performance after 200 epochs of training.

Model	Accuracy	Dice	F1-Score	Jaccard	Precision	Recall
UNet	0.784	0.784	0.781	0.659	0.784	0.784
Squeeze-UNet	0.691	0.691	0.658	0.539	0.635	0.691
DenseNet-67	**0.820**	**0.820**	**0.816**	**0.705**	**0.817**	**0.820**
DenseNet-56	0.789	0.789	0.782	0.663	0.789	0.789

With models trained on 320×240 images for 200 epochs, results are very close (Table 5). While UNet has the smallest process (Table 4), it is still relatively big. On the other hand, DenseNet-67 obtains the best performance scores on all metrics. Also we notice only a 0.031 drop in Dice score with the DenseNet-56 which shows that smaller models are able to perform just as well for semantic segmentation with a small number of classes dataset. This proves that for our task, resource-efficient models have great potential and could have industrial use. The confusion matrix (Figure 3) shows an imbalance in the model's perfor-

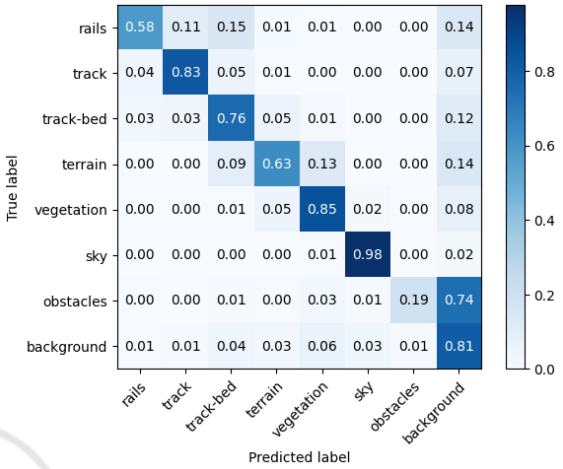

Figure 3: DenseNet-67 normalized confusion matrix.

mance. If cross-referenced with Table 1, we can see that the model clearly performs much better in classes that are well represented, but has more trouble with underrepresented classes like obstacles that are only in 13.8% to 24.4% of the frames in the dataset. This highlights the weakness of smaller models in their capacity to generalize for rare objects.

5.2 Quantization

In this work, we further explored model scaling through the quantization of models. Quantization compresses weights in float32 towards a smaller size, for instance, int8. We tested it under two scenarios post-training dynamic quantization (PTDQ) and quantization-aware training (QAT). PTDQ is implemented after training, truncating the values obtained with a clipping range calibrated for each input. As a result, the model loses in precision and performance. On the other hand, QAT intervenes during training in order to obtain a representation of the float operations with lesser precision thus including it in the loss calculations. PTDQ is easily implemented on a GPU, but QAT is not supported to this day for GPU. As a result, it only performs on CPU, the model's inference is therefore a concern. Because of ecological ethics, the model with QAT was halted after 1 epoch as it lasted 50 min with the resources available. Table 6 shows the results of our different scenarios.

We decided to focus mainly on our most

Table 6: DNet-56 before/after quantization (100 epochs).

Model	Accuracy	Dice	F1-Score	Jaccard	Precision	Recall
DenseNet-56	0.753	0.753	0.744	0.617	0.754	0.753
DNet-56 (PTDQ)	0.701	0.701	0.691	0.551	0.690	0.701
DNet-56 (QAT)	NA	NA	NA	NA	NA	NA

lightweight model, DenseNet-56, for quantization. The difference in performance between the regular and PTDQ version is only of 0.052 for the Dice score. Again, it shows potential in terms of margin of error.

5.3 Task Evaluation

We implemented a comparative study between different datasets in order to test the usefulness of our work for other similar segmentation tasks. As Cityscapes inspired RailSem19, it was chosen for our case study. When tested on the original Cityscapes dataset that contains 35 classes, DenseNets obtains very poor performance results (Table 7), close to random. From these results, we can infer that a 35 segmentation class task is too complex for the small architectures we have implemented. To verify this fact, we remapped Cityscapes to match RailSem8 (Table 8) to effectively compare performances on these datasets.

Table 7: Dice score on different datasets (200 epochs).

Model	Cityscapes 35	Cityscapes 8	RailSem8
UNet	NA	0.776	0.784
DenseNet-56	0.556	0.741	0.789
DenseNet-67	**0.584**	**0.796**	**0.820**

As the represented subject is different, class proportions are also vastly unmatched. Cityscapes is built with images taken from a vehicle ego-point of view, but set in a different context. Where RailSem8 represents mostly rails surrounded by nature, Cityscapes shows urban areas with a much higher concentration of humans, cars or other types of vehicles, all considered as obstacles. After modifying Cityscapes, we notice a considerable increase in performance of our models (Table 7). Results on Cityscapes8 are lower than RailSem8, but with a maximum drop of 0.06 in Dice score for the DenseNet-56 and minimal drop of 0.008 for the UNet. The similitude in tasks explains the closeness in results. Another aspect is that the label policy put in place for RailSem19 is based of the one from Cityscapes. As a result, the dense annotation is similar in style and approximation. On the other hand, Cityscapes counts 5000 images where RailSem19 has 8500, this accounts for a slightly better generalization of the models using RailSem19 which, in turn, explains the small performance drop. As a result, we can conclude that our models can efficiently perform segmentation for a low number of classes while being insufficient for more detailed tasks.

6 DISCUSSION

This work towards efficient and lightweight deep learning for semantic segmentation of railway environments has shown promise, but there are still limitations in terms of performance. While we were able to demonstrate potential in the use of light models for segmentation, it is paramount to appreciate the fact that up-scaling the model size is not the only way to enhance performance. For instance, a closer look at segmentation results after DPTQ shows a slight increase in performances for the detection of underrepresented classes in our dataset. Moreover, we observed a drop in performance for the detection of terrain, but an increase for the rail class. This observation shows that there is still room for optimization in the computing of feature maps that could influence the training process as a whole in favor of smaller classes. Future work might include the exploration of the effects of quantization on these values and the model's behavior regarding feature map construction for these classes.

For resource-efficient processing, smaller models do not always mean less resources needed (Table 4). As a result, a model with more FLOPs and parameters like UNet can show lower computational complexity while performing similarly if not slightly better. For an industrial application, the systems needs to be implementable on-board in trains. As a result, a slightly bigger model like UNet could still prove more suited to commercial use as it is closer to real-time use with less costly equipment. Therefore, with the available data, we consider UNet to be the most appropriate solution for industrial use in a constrained system. The skip connection mechanisms employed in the UNet and DenseNet architectures are at the root of the difference in operation processes. All in all, UNet is deeper in terms of number of convolution layers but faster overall. This problematic shows the challenge of balancing model and learning process frugality optimization.

Our work has proven that a lower resolution can show promising results without compromising learning efficiency. Naturally, there is a great chance that these models could perform better on higher resolution images, but that would intricately increase process requirements and therefore clash with our constraints. Future works include the implementation of attention mechanisms or data augmentation to enhance segmentation performance on less represented classes such as tracks. We also plan to implement this section of our research in a more global vegetation assessment application for railway maintenance.

Table 8: Cityscapes (CS 35) modified dense labels to Cityscapes8 (CS 8) compared to RailSem8 (RS8) labels.

Label RS8	Rails	Track	Track-bed	Terrain	Vegetation	Sky	Obstacles	Background
Label CS 8	Void	Tracks	Road	Terrain	Vegetation	Sky	Obstacles	Background
Label CS 35	-	rail track	road	terrain	vegetation	sky	person, rider, car, truck, bus, caravan, trailer, motorcycle, bicycle, license plate	unlabeled, ego vehicle, rectification border, out of roi, static, dynamic, ground, sidewalk, parking, building, wall, fence, guard rail, bridge, tunnel, pole, polegroup, traffic light, traffic sign

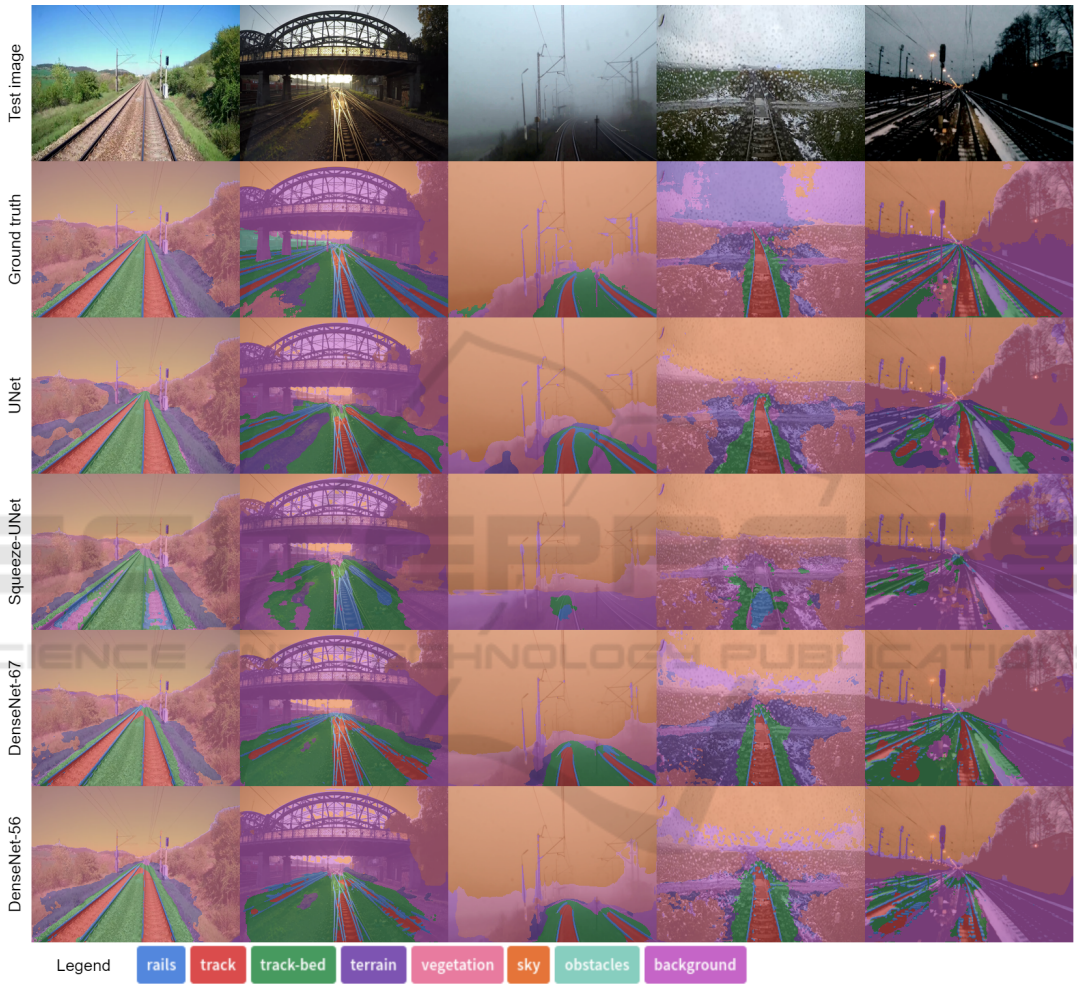

Figure 4: Output of each model after training for 200 epochs.

7 CONCLUSION

In this paper, we propose a comparative study for resource-efficient CNNs for semantic segmentation. The models presented consist of the implementation of known scaling techniques such as skip connections, and the adaptation of classifiers for segmentation tasks while making the most of the mechanisms developed to reduce model size. In our work, we constrained our models to smaller architectures while evaluating their performances on RailSem19 modified to RailSem8 and Cityscapes with 35 and 8 classes. Results show that our models have a very small drop of performance when quantized and are capable of detecting the main regions of interest for industrial use in the detection of vegetation. In the future, we will focus on optimizing results in order to achieve real-time performance and raise segmentation accuracy for under-represented classes.

REFERENCES

Ahamad, A., Sun, C.-C., and Kuo, W.-K. (2022). Quantized Semantic Segmentation Deep Architecture for Deployment on an Edge Computing Device for Image Segmentation. *Electronics*, 11(21):3561.

Badrinarayanan, V., Kendall, A., and Cipolla, R. (2017). SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(12):2481–2495.

Beheshti, N. and Johnsson, L. (2020). Squeeze U-Net: A Memory and Energy Efficient Image Segmentation Network. In *2020 IEEE CVPR Workshops*, pages 1495–1504.

Duquene, A. P. (2023). *Apprentissage machine pour la décision de conduite autonome de véhicules guidés : Application dans le domaine ferroviaire*. PhD thesis, Université Polytechnique Hauts-de-France.

Evchenko, M., Vanschoren, J., Hoos, H. H., Schoenauer, M., and Sebag, M. (2021). Frugal Machine Learning. *arXiv:2111.03731*.

Harb, J., Rébéna, N., Chosidow, R., Roblin, G., Potarusov, R., and Hajri, H. (2020). FRSign: A Large-Scale Traffic Light Dataset for Autonomous Trains. *arXiv:2002.05665*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *2016 IEEE CVPR*, pages 770–778.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. In *2017 IEEE CVPR*, pages 2261–2269.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and $\leq$ 0.5mb model size. *arXiv:1602.07360*.

Li, X. and Peng, X. (2023). Rail Detection: An Efficient Row-based Network and A New Benchmark. *ACMM 2022 arXiv:2304.05667*.

Nanfack, G., Elhassouny, A., and Thami, R. O. H. (2017). Squeeze-SegNet: A new fast Deep Convolutional Neural Network for Semantic Segmentation. *ICMV 2017*.

Pappaterra, M. J., Flammini, F., Vittorini, V., and Bešinović, N. (2021). A Systematic Review of Artificial Intelligence Public Datasets for Railway Applications. *Infrastructures*, 6(10):136.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *MICCAI 2015 arXiv:1505.04597*.

Singh, P., Dulebenets, M. A., Pasha, J., Gonzalez, E. D. R. S., Lau, Y.-Y., and Kampmann, R. (2021). Deployment of Autonomous Trains in Rail Transportation: Current Trends and Existing Challenges. *IEEE Access*, 9:91427–91461.

Skibicki, J. D. and Licow, R. (2022). A Visual Method of Measuring Railway-Track Weed Infestation Level. *Metrology*, 2(2):230–240.

Tan, M. and Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *ICML 2019*, pages 6105–6114.

Wang, Y., Wang, L., Hu, Y. H., and Qiu, J. (2019). RailNet: A Segmentation Network for Railroad Detection. *IEEE Access*, 7:143772–143779.

Yin, M., Li, K., and Cheng, X. (2020). A review on artificial intelligence in high-speed rail. *Transportation Safety and Environment*, 2(4):247–259.

Yvinec, E. (2023). *Efficient Neural Networks : Post Training Pruning and Quantization*. PhD thesis, Sorbonne Université.

Zakaria, B., Ben Ahmed, O., Amamra, A., Bradai, A., and Beghdad Bey, K. (2022). PSCS-Net: Perception Optimized Image Reconstruction Network for Autonomous Driving Systems. *IEEE ITS*, 24(2):1–16.

Zendel, O., Murschitz, M., Zeilinger, M., Steininger, D., Abbasi, S., and Beleznai, C. (2019). RailSem19: A Dataset for Semantic Rail Scene Understanding. In *2019 IEEE CVPR Workshops*, pages 1221–1229.

Zouaoui, A., Mahtani, A., Hadded, M. A., Ambellouis, S., Boonaert, J., and Wannous, H. (2022). RailSet: A Unique Dataset for Railway Anomaly Detection. In *2022 IEEE IPAS*, pages 1–6.