Deep Learning-based Rail Surface Condition Evaluation Supplementary Material

Shilin Hu[1], Ke Ma[2], Sagnik Das[1], Dichang Zhang[1], Dimitris Samaras[1]
[1]Stony Brook University [2]Snap Inc.
{shilhu, kemma, sadas, diczhang, samaras}@cs.stonybrook.edu

In this supplementary material, we provide the following:
- Severity level definitions used by expert annotators
- Additional details on segmentation and alignment results
- Classification module backbone comparisons and data augmentation analyses

1. Severity Level Annotation Criteria

We adopt a standardized 8-level severity scale defined and annotated by domain experts, with levels ranging from 0 (no defect) to 7 (severe surface deterioration). These levels are based on observable surface features, including defect geometry, visual prominence, and material degradation:
- **Level 0**: No visible surface defects.
- **Level 1**: Barely perceptible surface marks with clearly regular patterning.
- **Level 2**: Clear, distinct surface cracks without pitting at the crack tip.
- **Level 3**: Visible cracking with small pits up to 4 mm in diameter.
- **Level 4**: Pitting with a diameter greater than 4 mm but less than 10 mm.
- **Level 5**: Isolated pitting, shelling, or spalling with a diameter greater than 10 mm and depth up to 5 mm.
- **Level 6**: Shelling, spalling, and consistent pitting greater than 10 mm in diameter.
- **Level 7**: Severe shelling/spalling and any defect greater than 16 mm in diameter and longer than 20 mm.

This expert-defined severity scale not only enables consistent training and evaluation of the framework but also directly supports rail maintenance decision-making by providing actionable defect categories for inspection prioritization and repair planning.

2. Segmentation and Alignment on the Proprietary Dataset

2.1. Dataset Description

The proprietary dataset contains 939 rail images captured by the RailScope platform [8] across various active service tracks in North America. Each image is accompanied by an expert-labeled rail surface mask, which is used to train and evaluate the segmentation and alignment modules.

Table 1. Comparison of rail surface segmentation results. We report the mean and standard deviation of the Jaccard Index, model parameter count, and inference time. $*$ indicates the inference time reported in [10]. Best results highlighted in **bold**.

Method	Mean JI	Std JI	# Params (M)	Inference Time (s)
Baseline[10]	0.986	0.031	-	4.8*
7-layer U-Net	0.985	0.012	42	0.012
8-layer U-Net	0.987	0.012	54	0.015
Ours	**0.991**	**0.010**	**2**	**0.015**

2.2. Segmentation Results

We assess the performance of our segmentation module using the Jaccard Index (JI):

$$J(R_s, R_g) = \frac{|R_s \bigcap R_g|}{|R_s \bigcup R_g|}, \tag{1}$$

where R_s is the predicted rail surface region and R_g is the ground truth rail surface mask.

Table 1 presents a comparison of our MobileNetV2-ASPP-based segmentation module, U-Net baselines, and the structured random forest baseline from [10]. Our model achieves a mean JI of 0.991 and a standard deviation of 0.010, outperforming all baselines both in accuracy and consistency. The low standard deviation ($0.031 \rightarrow 0.010$, a 67% reduction) reflects significantly improved prediction stability across diverse track conditions.

In terms of model complexity, our module contains only 2 million parameters, reducing model size by over 95% compared to the U-Net variants. Additionally, the inference time is over 48 times faster than the CPU-based baseline, owing to GPU-accelerated deep learning.

Fig. 2 further compares segmentation outputs visually. The MobileNet-ASPP model produces cleaner, straighter rail boundaries and fewer artifacts compared to U-Net,

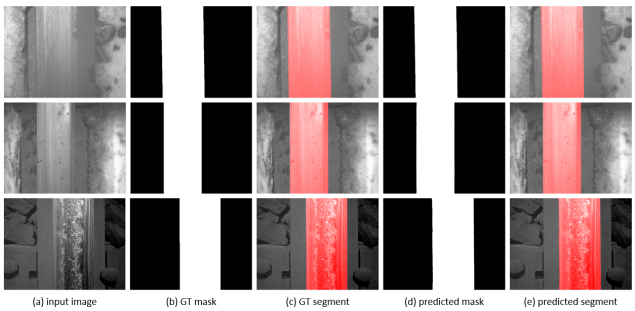

Figure 1. Examples of segmentation results. (a), (b) and (d) show input images, ground truth masks, and predicted masks, respectively. (c) and (e) overlay the masks onto the original images. Results are produced by the proposed module with input resolution 512×512, upsampled to 1200×1600.

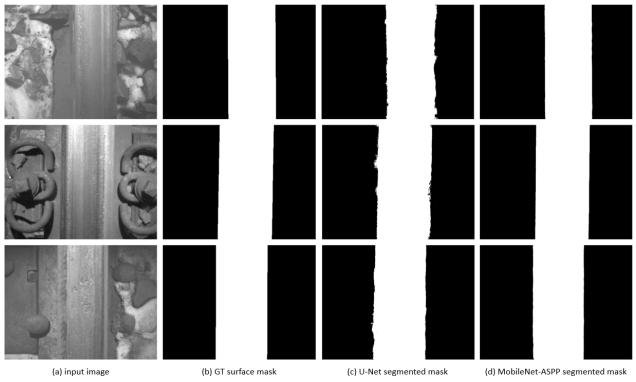

Figure 2. Qualitative comparison of segmentation results. (a) shows input images, (b) are ground truth masks, (c) are U-Net outputs, and (d) are from the proposed MobileNet-ASPP model [11].

which exhibits larger deviations from ground truth masks. These results reinforce the effectiveness of our lightweight and accurate segmentation design.

2.3. Alignment Results

Unlike the baseline method, which relies on hard-coded rail surface edge alignment, our approach introduces a learned alignment module that predicts an affine transformation to reposition the rail surface to a canonical view.

We first evaluate the alignment module using ground truth masks to extract rail surface segments. As shown in Tab. 2, this setup achieves a mean Jaccard Index (JI) of 0.983 with a standard deviation of 0.024. Fig. 3 shows qualitative results, where the module effectively aligns the rail surface to the image center. Surprisingly, alignment performance improves when using predicted segmentation masks instead of ground truth masks. This indicates that our combined training allows the segmentation and alignment modules to co-adapt, resulting in better downstream quality. Additionally, we find that using segmentation masks from the

Table 2. Rail surface alignment performance using different segmentation masks.

Segmentation Source	Mean JI	Std JI
Ground Truth Masks	0.983	0.024
U-Net	0.972	0.012
MobileNet-ASPP	**0.989**	**0.007**

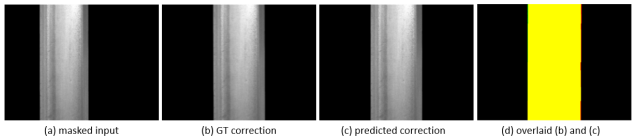

Figure 3. Examples of alignment results. (a) masked rail surface, (b) ground truth aligned region, (c) our module output, (d) overlay of (b) in **green** and (c) in **red**; **yellow** indicates intersection.

Table 3. RMSE comparison between final alignment outputs and ground truth segments.

Method	RMSE
Repositioned masked segment	7.09
Transformed original image	**7.00**

U-Net model results in a lower alignment score (mean JI of 0.972), suggesting that the alignment module is sensitive to the quality of the segmentation output.

To further assess the impact of how the transformation is applied, we compare two options: directly using the repositioned masked rail segment or applying the predicted transformation matrix to the original image. We compute the Root Mean Square Error (RMSE) between these outputs and the ground truth rail surface segments. As shown in Tab. 3, applying the transformation to the original image yields a lower RMSE, indicating more accurate spatial alignment and fewer inherited artifacts.

To avoid propagation of segmentation errors, we apply the predicted affine transformation directly to the original image rather than to the masked rail surface. This preserves finer surface details and mitigates artifacts introduced by the segmentation masks. As illustrated in Fig. 4, directly using the repositioned masked segment can inherit noise or boundary errors, while applying the transformation to the original image yields more accurate and cleaner alignment results for downstream classification.

3. Classification Module

3.1. Model Selection

We benchmark a variety of image classification architectures, including ResNet, DenseNet, EfficientNetV2, MobileNet, Swin Transformer, and Vision Transformers (ViT),

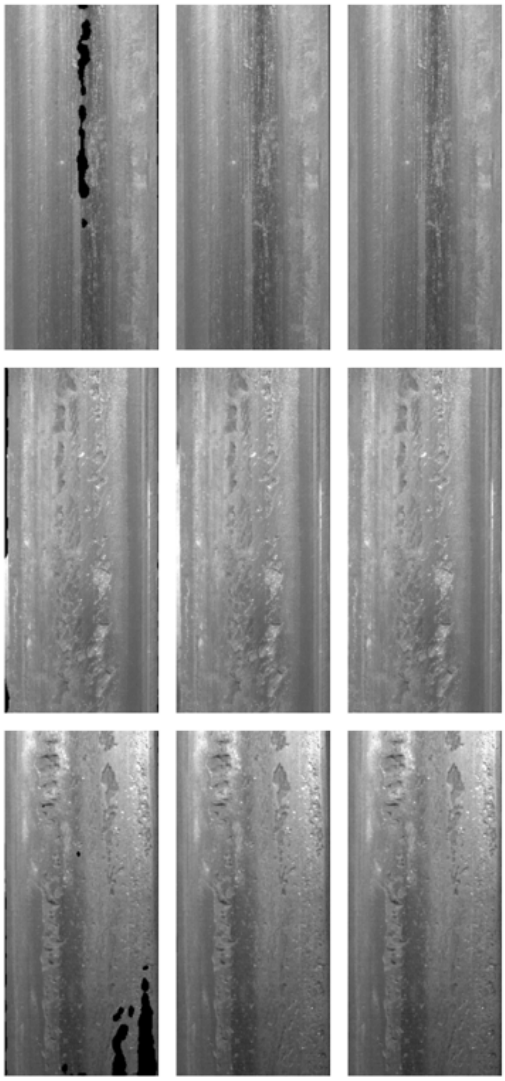

Figure 4. Alignment output comparison. (a) repositioned masked rail surface, (b) transformed original image segment, (c) ground truth rail segment.

to assess their suitability for the rail surface severity classification task. All models are initialized with ImageNet [2] pre-trained weights and fine-tuned on our dataset.

Tab. 4 reports each model's classification accuracy and inference speed. While VGG16 achieves the highest accuracy at 80.74%, and Swin-S follows closely with 80.57%, both models fall behind in efficiency. In contrast, ResNet18 delivers the fastest inference speed at 95 FPS and maintains a competitive accuracy of 79.06%, just 1–2% lower than the top models. We observe that deeper or more complex architectures (e.g., ResNet50+, DenseNet, and Vision Transformers) do not yield substantial accuracy gains. This suggests that the rail surface dataset does not benefit significantly from deeper representational capacity.

Table 4. Rail Surface Classification Results: Prediction accuracy and inference speed across different models. Best results are in **bold**, second best are underlined.

Model	Accuracy (%)	Infer. speed (FPS)
VGG16[13]	**80.74**	46
VGG19[13]	79.42	42
ResNet18[5]	79.06	**95**
ResNet34[5]	80.21	77
ResNet50[5]	77.39	58
ResNet101[5]	78.45	40
DenseNet121[7]	77.65	34
ResNeXt50[15]	77.65	41
ResNeXt101[15]	74.91	37
MobileNetV2[12]	76.59	74
MobileNetV3S[6]	77.92	73
MobileNetV3L[6]	78.45	67
EfficientNetV2s[14]	80.21	31
ViT[3]	69.88	70
Swin-T[9]	79.15	47
Swin-S[9]	80.57	28

Table 5. Classification accuracy using different data augmentation strategies.

Augmentation Method	Accuracy (%)
Ours w/o Aug.	79.06
+ Pixel-level Aug.	**81.80**
+ Mixup[17]	79.86
+ CutMix[16]	81.18
+ FMix[4]	77.92
+ GridMask[1]	79.59

Given the relatively narrow spread in accuracy across models and our emphasis on efficiency, we select ResNet18 as the classification backbone for our final framework due to its favorable balance between speed and accuracy.

3.2. Data Augmentation Analysis

To enhance the robustness and generalization of our classification model, we evaluate several data augmentation strategies. These include pixel-level augmentations (brightness, contrast, Gaussian blur, ISO noise, and JPEG compression) and mixed sample data augmentation (MSDA) methods such as Mixup [17], CutMix [16], FMix [4], and GridMask [1].

Tab. 5 presents the results. Pixel-level augmentations yield the most significant improvement, increasing accuracy by 2.74% over the baseline (from 79.06% to 81.80%).

Among MSDA methods, CutMix achieves the highest accuracy (81.18%), but still falls short of pixel-level augmentation. Mixup and GridMask offer only marginal improvements, while FMix reduces performance, likely due to excessive structural distortion of defect features.

Based on these results, we adopt pixel-level augmentations as our default strategy for training the final classification module.

References

[1] Pengguang Chen, Shu Liu, Hengshuang Zhao, and Jiaya Jia. Gridmask data augmentation, 2020. 3

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 3

[3] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. 3

[4] Ethan Harris, Antonia Marcu, Matthew Painter, Mahesan Niranjan, Adam Prügel-Bennett, and Jonathon Hare. Fmix: Enhancing mixed sample data augmentation, 2021. 3

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[6] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1314–1324, 2019. 3

[7] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 3

[8] KLD Labs Inc. Rail surface assessment — railscope™ system. https://www.kldlabs.com/track-assessment-2/rail-surface-assessment/, 2025. 1

[9] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 3

[10] Ke Ma, Tomás F Yago Vicente, Dimitris Samaras, Michael Petrucci, and Daniel L Magnus. Texture classification for rail surface condition evaluation. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016. 1

[11] A. Milioto and C. Stachniss. Bonnet: An Open-Source Training and Deployment Framework for Semantic Segmentation in Robotics using CNNs. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019. 2

[12] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 3

[13] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 3

[14] Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *International conference on machine learning*, pages 10096–10106. PMLR, 2021. 3

[15] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks, 2017. 3

[16] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features, 2019. 3

[17] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization, 2018. 3