Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 00 (2026) 000–000

Transportation Research Procedia

www.elsevier.com/locate/procedia

World Conference of Transport Research, Toulouse 2026 (WCTR 2026 Toulouse)

# Weak supervision and fine-tuning with contrastive learning for multiclass lane marking segmentation

Júlia Alves Porto[a]*, Apostolos Ziakopoulos[a], George Yannis[a]

[a]*National Technical University of Athens, 5 Heroon Polytechniou Str, Athens, GR-15773, Greece*

**Abstract**

Lane delineation is essential information for autonomous vehicles. Being able to further differentiate between lane marking types gives the computational system a greater level of information regarding the road system and could improve safety assessment relying on infrastructure data and infrastructure management by itself. In this work, we propose a dual-branch weakly supervised pipeline for lane marking segmentation, relying on a Bootstrap Your Own Latent – BYOL self-supervised branch that uses contrastive loss to learn intrinsic patterns from the training dataset and a LinkNet segmentation branch for the lane marking segmentation itself; alternatively, BYOL self-supervision is used to fine-tune a pretrained segmentation model. The open dataset TuSimple is used as the base dataset for our analysis, and lane marking types are manually assigned after the labeled pixels are separated into individual entities based on their graph structure, forming an annotated ground truth dataset with noisy labels. Results show that fine-tuning with contrastive learning increases the semantic information learned by the segmentation model, leveraging more accurate lane delineation extraction, while weak supervision presents consistent learning but lower metric scoring. This work presents a novel discussion on the application of self-supervision for multiclass lane delineation with potential application in transport safety and management.



Peer review under the responsibility of the World Conference on Transport Research – WCTR 2026

*Keywords:* lane delineation; weak supervision, contrastive loss, lane marking types; segmentation

## 1. Introduction

Lane delineation is a very important asset for autonomous vehicles (AV), as it is imperative that the vehicle obeys the expected behavior on a road. Not only can new manufactured vehicles already include lane-assisting technology, but also computer vision algorithms have been trained on lane delineation datasets achieving high performance results (Hoang et al., 2016; Khan et al., 2025; Merugu and Adarsh, 2022; Nie et al., 2025; Zoljodi et al., 2024).

* Corresponding author. Tel.: 30-694-996-1497; +55-61-98128-8285.
*E-mail address:* julia_porto@mail.ntua.gr



Peer review under the responsibility of the World Conference on Transport Research – WCTR 2026

From a safety point of view, further information about the lane marking types and quality can provide meaningful information relatable to driving performance. Sharmin et al. (2023), for example, found that even careful-impaired drivers tend to perform poorly when driving under adverse conditions, including when no lake markings are present. The absence of lane markings on asphalt pavement typically results from either deterioration or, in newly constructed or resurfaced pavements, delays by the responsible authority in completing the markings before opening the road to traffic. For concrete pavement, some road authorities rely on expansion joints between concrete slabs to indicate lane boundaries.

A common practice to automate quality evaluation of lane markings is a two-stage approach: first, a CV-based model identifies the lane marking object and its limits, than the actual paint limits are compared with what a fully painted, good quality lane marking would look like, either by comparing the final shape or the RGB contrast between light colours (white, yellow) for the lane markings and dark colours (gray, black) for the pavement (Kong et al., 2022; Mi et al., 2021). However, in the case of lane delineation, the existence of dashed markings means that this analysis is subject to the identification of the individually painted segments as separate entities, making it more subject to image distortion. We believe that identifying different lane types can enhance quality assessments and provide continuous contextual information on road regulations within a segment, such as channelization and permitted turning movements.

Furthermore, although lane delineation has been extensively studied, lane marking types have not received comparable attention from the academic community, even when using state-of-the-art imaging solutions such as LiDAR (Gargoum and El-Basyouny, 2017). Only a limited number of examples exist in the literature, while recent reviews have showed that the most comprehensive and commonly used datasets provide binary labels for road markings or, at most, lane counts based on delineation, without further differentiation of lane type (Mamun et al., 2022). Meanwhile, advances in artificial intelligence (AI) and global connectivity enable further enhancement of existing models in toward faster and more sustainable algorithms.

In this work, we train a dual-branch weakly supervised model for lane delineation as a multiclass task, dividing lane markings into three types: continuous, dashed and unmarked. Our proposed pipeline relies on state-of-the-art models: LinkNet for the segmentation branch, a high performing and light weight segmentation model with multiple applications; and Bootstrap Your Own Latent – BYOL for the self-supervised branch, a learning method that relies on architectural asymmetry and contrastive loss to learn intrinsic representation patterns from the input dataset. Weak supervision has already been shown to improve lane delineation segmentation by previous research. In the present research, additional novelty is introduced by turning the TuSimple dataset into a multiclass dataset, leveraging an additional layer of difficulty for the model with extra information of the road environment. The aim is to contrast the performance of our proposed approach with full supervision using the same LinkNet backbone model and with BYOL fine-tuning under different augmentation and dataset configurations. Therefore, this work has three novel contributions to the scientific community:

- Adding a self-supervised learning branch to road delineation segmentation without the use of negative examples.
- Showing how self-supervision helps the segmentation model to learn under different configurations, with special impact when used in combination with full supervision for model fine tuning.
- Publishing a new, open-source multiclass lane marking dataset.

Apart from this general introduction and objective description (Section 1), the rest of the paper is organized as follows: Section 2 presents related studies, focusing on lane marking segmentation, contrastive learning and weak supervision general applications, and lane marking segmentation via contrastive learning; Section 3 describes the methodology applied in the present research and the dataset used for model training; Section 4 presents the results and the associated discussion that the results present; and Section 5 summarizes the research outputs, i.e., conclusions, limitations and future work.

## 2. Related work

### *2.1. Lane marking segmentation*

Lane delineation has often very characteristic shape and colour features, allowing for computer vision automated detection even before popularization of deep learning algorithms, as proposed by Chiu and Lin (2005). During more recent years, the use of deep learning has become quite common for producing very accurate detections capable of succeeding in more challenging scenarios (Mamun et al., 2022), often combined with non-AI post-processing (Tran and Le, 2019) or pre-processing (Muthalagu et al., 2021). Mamun et al. (2022) highlight that, among Deep Learning-based extraction techniques, semantic segmentation seems to be the most appropriate given the structure of lane delineation and, for end-to-end segmentation approaches, common encoder-decoder architectures achieve successful detection rates, even more so when combined with dilated convolution modules and pooling operations.

Deep learning can also be used in addition to shape cues, as in the case of CLRNet (Zheng et al., 2022), where, after high-level lane features detection is made through a traditional encoder, a convolutional network termed ROIGather is used to identify the Region of Interest (ROI) of the identified lane pixels and to sample them with bilinear interpolation. Despite having trained high-performance models, researchers still find room for accuracy improvement, speed performance increase and addressing detection techniques under challenging scenarios (Zakaria et al., 2023).

Only a few papers have addressed lane marking type extraction. Hoang et al. (2016) proposed discriminating between continuous and dashed markings for road lane delineation by applying a post-processing pipeline after lane detection that leverages the line segments angles, inter-distance and position. More recently, Merugu and Adarsh (2022) have also pursued lane delineation with different type classification using a Convolutional Neural Network improved by a curve fitting mechanism. Their best performing model achieved 66% F1-Score and 88% IoU on TuSimple dataset, although it is not clear whether those metrics are considering the different lane types existing in the data.

### *2.2. Contrastive learning and weak supervision*

Contrastive learning is a type of self-supervised representation learning that works by approaching different representations of the same image on the embedding space. It is usually based on augmented pairs of positive samples trained in contrast with negative samples, i.e., images that do not correspond to an augmented version of the target image. For example, SimCLR (Chen et al., 2020) applies strong data augmentations to create positive pairs and uses a contrastive loss function over a large set of negatives. MoCo (He et al., 2020) improves efficiency by introducing a dynamic memory bank to store negative examples, enabling contrastive learning with smaller batch sizes. More recently, BYOL (Grill et al., 2020) challenged the necessity of negatives altogether by aligning an online network with a slowly updated target network, relying on architectural asymmetry to avoid collapse. These methods have demonstrated strong performance in learning transferable representations without labels (Dippel et al., 2022).

Weak supervision refers to training paradigms that rely on limited or imperfectly labeled data, often supplemented with self-supervised signals or noisy annotations. Instead of relying purely on fully annotated datasets, weak supervision leverages auxiliary cues to improve generalization. For example, Meletis et al. (2019) use two weak supervision approaches to select the most relevant data to enhance semantic segmentation performance: first one is to find similar images using a Gaussian Mixture Model (GMM); second one is to find images with high object diversity by relying on bounding box labels. Their method was capable to optimize performance by reducing the number of required labels for training by 20 times on Cityscapes data. In turn, Löwens et al. (2023) propose the use of programmatic weak supervision in addition with self-supervision embeddings to create weak, noisy Stay Region (SR) labels, i.e., regions where the road user spend a considerable amount of time, and use these labels to train a transformer-based encoder-decoder model that classifies points within a trajectory as belonging to a stay or non-stay area. Using self-reported activities of the “ExtraSensory” dataset as ground truth labels, their results outperformed prediction metrics of similar works that use solely unsupervised models to classify SR.

### *2.3. Lane marking segmentation with contrastive learning*

Recent research has shown that contrastive learning can improve model performance for lane delineation in challenging scenarios, such as shadows and crowded scenes. Zoljodi et al. (2024) proposed a self-supervised learning method named Contrastive Learning for Lance Detection via Cross-Similarity (CLLD) that, as the name suggests, uses cross-similarity contrastive loss function to assess local similarities within the global context of the input image during training. Their proposed loss function improves U-Net recall and F1-Score performance on CuLane dataset and accuracy on TuSimple dataset. Nie et al. (2025) proposed LaneCorrect, a self-supervised training pipeline trained with LiDAR data that performs comparably to fully supervised models trained on benchmark datasets. Khan et al. (2025) proposes the use of contrastive learning to fine-tune lane detection models and shows a better embedding for different datasets under their proposed pipeline. Therefore, it is apparent that there are knowledge gaps in the literature regarding automated lane marking type extraction, even though this road feature appears to be critical for improving road safety performance, especially regarding automated vehicles.

## 3. Methodology

### *3.1. Dataset preparation*

TuSimple is an open-access dataset with images from US highways for lane detection challenges (Yoo et al., 2020). The dataset consists of approximately 6,000 folders, each folder containing at least 20 frames of images recorded with 1280x720 resolution size, taken from the driver's perspective of the road. For each frame, there is a corresponding json file with the lane delimitation description, which can be used to build the semantic segmentation masks. To avoid data leakage, only one image from each folder was selected to construct the TuSimple subset.

For the lane marking type classification, each lane instance from the ground truth labels had to be separated into a single object. With this purpose, the following steps were taken: (i) thinning of segmentation mask pixels; (ii) building of a graph structure from 8-level connectivity using NetworkX library; (iii) identification of graph's endpoints, i.e., pixels with connectivity on only one direction; (iv) classification of separate entities by assessing the number of connections, if the graph had more than two connections, a new entity was defined. Finally, the separate graph entities were used as markers at SciPy's Watershed function, to maintain simultaneously the individual entities separation and the original lane thickness.

This graph-based pipeline was selected to separate individual lane markings where pixels overlapped and selected over tests on a frame with multiple overlapping pixels. Nevertheless, it is not fail-proof and some lanes of different types remained assigned to a single entity. Furthermore, lane type assignments were done manually and subject to human error. Therefore, the TuSimple subset with lane marking types is also considered a dataset with noisy labels. Finally, the partition of the dataset for training, test and validation followed a 7:1:2 distribution and contains: 4,485 image and mask pairs for training; 641 pairs for validation and 1,2828 for testing. A similar distribution of lane types per partition was assured, as can be seen by Table 1.

Table 1. Mean proportion of pixels for each lane marking type per dataset partition.

| Marking type | Overall (%) | Train (%) | Validation (%) | Test (%) |
|---|---|---|---|---|
| Continuous | 0.8976 | 0.8990 | 0.8949 | 0.8939 |
| Dashed | 1.2352 | 1.2299 | 1.2451 | 1.2491 |
| Unmarked | 0.5776 | 0.5790 | 0.5782 | 0.5725 |

### *3.2. Segmentation model*

For the segmentation model, LinkNet architecture was selected for having state-of-the-art results in other segmentation tasks and low computational costs when compared to similar popular models (Porto et al., 2025).The LinkNet architecture consists of a encoder-decoder architecture, where the results of each encoder block is added to the corresponding layer on the decoder-block, allowing to maintain spatial information that otherwise would have

been lost in the downsizing initial part (Chaurasia and Culurciello, 2017). Residual blocks are also used inside the encoder blocks. In this work, ResNet50 (He et al., 2016) is used as the encoder of the LinkNet model.

The Loss Function used for monitoring performance of the segmentation model was Cross Entropy Loss (Equation 1) with the logit values predicted by the last layer of the model, as recommended for multiclass classification tasks. Although we are working with a segmentation model, the aim is to correctly classify each lane instance, so the selected loss function was deemed appropriate.

$$L_{CE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}\left(y_{i,j}\cdot log\left(p_{i,j}\right)\right) \quad (1)$$

On the above equation, N is the number of samples; C is the number of classes; $y_{i,j}$ is 1 if class j is correct for sample I and 0 otherwise; and $p_{i,j}$ is model-predicted probability of sample I being in class j.

### *3.3. Bootstrap Your Own Latent - BYOL model*

For the self-supervised component of our model, we adopt the Bootstrap Your Own Latent (BYOL) framework (Grill et al., 2020). In BYOL, two differently augmented views per input are processed through two different branches, online and target, sharing similar but asymmetrical structures. The first one uses a series of weights θ, updated by gradient descent driven from the training loss, while the second one uses a moving exponential average of θ, ξ. On the online branch, the augmented view (v) goes through the representation layer ($y_\theta = f_\theta(v)$), then through a projection layer ($z_\theta = g_\theta(y_\theta)$) and finally through a prediction layer ($p_\theta$). The target network goes through the representation and projection layer only. Then, the loss is calculated by the mean squared error between L2-normalized online prediction and target projection, as described in Equation 2. Mathematically, this equation is equivalent to 2 minus twice the cosine similarity between the L2-normalised prediction $p_\theta(z_\theta)$ and target projection $z'_\xi$. Since both vectors are unit-normalized, it is also equivalent to their dot product.

$$L_{\theta,\xi} = \left\|\overline{p_\theta}(z_\theta) - \overline{z'}_\xi\right\|_2^2 = 2 - 2\cdot\frac{\langle p_\theta(z_\theta), z'_\xi\rangle}{\|p_\theta(z_\theta)\|_2\cdot\|z'_\xi\|_2} \quad (2)$$

In this case, the representation layer is the same encoder used in the segmentation model, ResNet50; and both the projection and prediction layers are two-layer Multilayer Perceptron (MLP) with one-dimensional Batch Normalization. Furthermore, both augmented views are passed independently through each branch (online and target), and the average loss between the two views is used as the final loss. Lastly, since the cosine similarity has a range of [-1, 1], the theoretical range of BYOL's loss is [0, 4]. More details on the underlying theory of BYOL can be found in Grill et al. (2020) and Figure 1.

### *3.4. Training pipeline and performance monitoring*

The training pipeline was composed of the following steps:

- For each image/mask pair from the input batch, three different random augmentations were applied to the image, generating view 1, view 2 and the segmentation input
- Each image passes through the same Encoder (ResNet-50 was selected), from where the features of the second-to-last layer are retained
- Self-supervised BYOL branch:
  - Both view 1 (online) and view 2 (target) encoder features are processed through a projector network, a multi-layer perceptron (MLP), which maps the high-dimensional encoder output into a lower-dimensional embedding space (projection space).
  - View 1 (online) passes through a further predictor network, another MLP, which expands the projected features into a hidden dimension and maps them back to the projection dimension. This transformation makes the online prediction comparable to the target projection, while also breaking symmetry between the online and target networks to stabilize training.

- Projector and predictor features are used to calculate the BYOL Loss, as defined by Equation 2.
- Segmentation LinkNet:
  - Encoder features from the segmentation input pass through the LinkNet decoder, using the default parameters from Segmentation Models Pytorch (Iakubovskii, 2019).
  - The Segmentation Loss as defined by Equation 1 is applied using the predicted mask logits and the ground truth labeled masks.
- The average loss is calculated between BYOL and Segmentation losses and directs the backpropagation for the network training.

Figure 1 better illustrates the pipeline described previously. The training duration was set to 100 epochs, with early stopping for no improvement after 20 consecutive epochs, and the best model was saved according to the best average loss calculated on the validation dataset. The base form of Adam algorithm was used for optimization with a learning rate of 0.001. Training and testing were performed locally on NVIDIA GeForce RTX-2080 GPU, Cuda version 12.6.

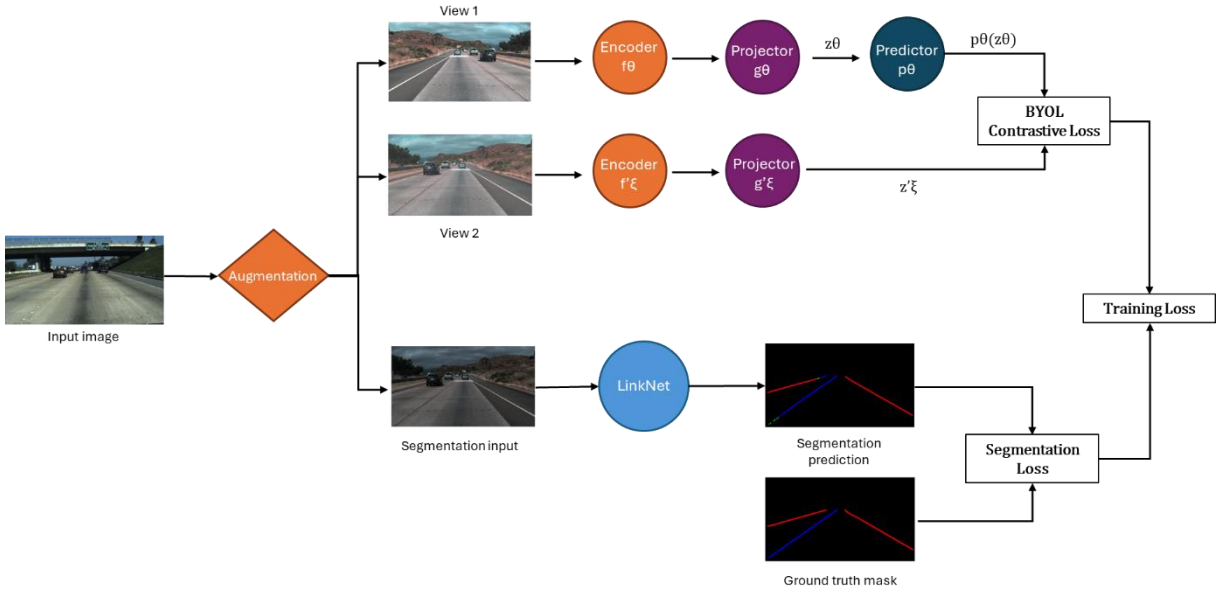



Fig. 1Training pipeline.

Training was performed according to the pipeline presented in Figure 1, but also with fine-tuning only. For fine-tuning, the LinkNet model was trained previously and the BYOL branch was applied afterwards to update encoder weights only.

Despite the presence of noise in the labels, standard segmentation metrics (F1-Score, Intersection of Union – IoU, Accuracy, Precision and Recall) were calculated for the test dataset to give further insights regarding the behaviour of the proposed pipeline. Equations 3 to 7 describe each metric.

$$F1 = 2 \times \frac{Precision \cdot Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{3}$$

$$IoU = \frac{P \cap T}{P \cup T} = \frac{TP}{TP + FP + FN} \tag{4}$$

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{5}$$

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

On the above Equations, TP are true positive pixels, FP are false positive pixels, TN are trye negative pixels and FN are false negative pixels. In the case of multiclass segmentation, metric values are calculated for each class individually.

## 4. Results and discussion

Due to memory availability, the training and validation batches were composed of 4 inputs only. Each epoch took approximately 2 hours to be trained, and the model achieved early stopping criteria after running for 87 epochs, which took over one week. Figure 2 shows the progress logs during the training process. The metrics were calculated only for the validation dataset, to facilitate transferability measurements.

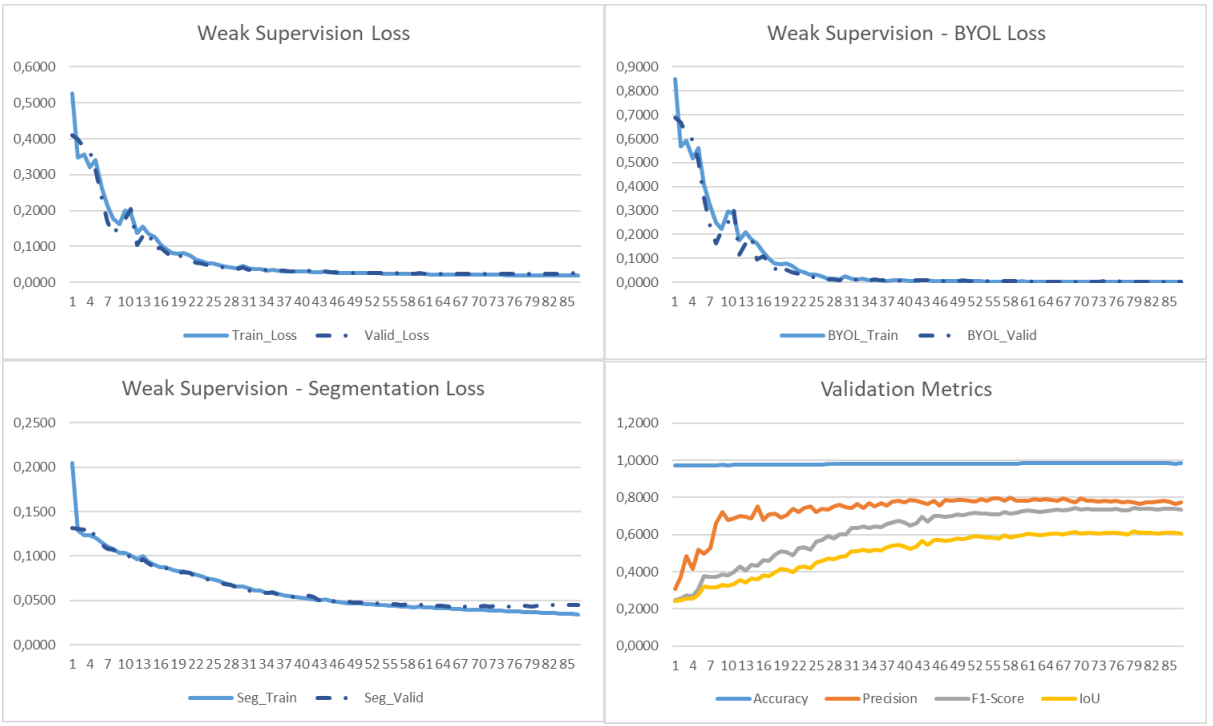



Fig. 2 Training logs.

Since the loss function for overall training is averaged between the two branch losses and the BYOL branch presented a slower conversion rate along with larger inter-epoch loss variations, the overall training behaviour reflected a similar pattern. It also seems that the training achieved a saturation point around the 30$^{th}$ epoch. After that, the decrease in the self-supervised branch loss is minimal compared to the earlier stages and, even though the segmentation branch loss continues decreasing, the validation loss stops accompanying it, indicating risk of over fitting.

The metrics measured indicate a similar behavior, as they increase substantially in the first 30 epochs and afterwards, only marginally. Accuracy remained high (over 95%) during all the training process, which highlights how frail it is as a performance indicator for this type of segmentation task. Table 2 presents the metric values calculated for the test dataset with our last saved model, from the 67$^{th}$ epoch. Inference time took approximately 15 minutes for 1121 batches, each with 4 images, giving an average rate of 0.8 seconds per batch.

Table 2. Metric results on the test dataset for the Weak Supervision pipeline.

| Marking type | F1-Score (%) | IoU (%) | Precision (%) | Recall (%) | Accuracy (%) |
|---|---|---|---|---|---|
| Average | 70.5451 | 59.2958 | 75.8919 | 70.7439 | 99.1930 |
| Background | 99.2125 | 98.4389 | 98.9700 | 99.4569 | 98.4652 |
| Continuous | 65.3877 | 53.9496 | 71.7200 | 71.3481 | 99.5016 |
| Dashed | 62.1712 | 50.9858 | 67.5384 | 71.7733 | 99.2028 |
| Unmarked | 63.6895 | 58.8667 | 69.7654 | 81.6769 | 99.6013 |

Although results seemed quite satisfactory, they did not exceed results achieved with only the Full Supervision (LinkNet) branch, as presented in Table 3. The LinkNet-only model was lighter, taking half the time for training per epoch, and converged much quicker (20$^{th}$ epoch) than the weakly supervised model.

Table 3. Metric results on the test dataset with LinkNet only.

| Marking type | F1-Score (%) | IoU (%) | Precision (%) | Recall (%) | Accuracy (%) |
|---|---|---|---|---|---|
| Average | 73.9580 | 63.0995 | 78.4964 | 74.0387 | 99.2659 |
| Background | 99.2845 | 98.5805 | 99.0978 | 99.4725 | 98.6061 |
| Continuous | 69.0686 | 57.7714 | 73.5414 | 74.3342 | 99.5225 |
| Dashed | 70.3468 | 59.5856 | 75.8446 | 74.1643 | 99.3011 |
| Unmarked | 72.6033 | 67.9909 | 78.9508 | 83.4450 | 99.6336 |

The failure to outperform the full supervision baseline is likely due to confusing signals from BYOL in end-to-end training. A similar result is observed by Porto et al. (2026) when applying the proposed approach to a binary segmentation task. The weak supervision approach, however, showed improvement for result transferability, especially when used as a fine-tuning step. Thus, we discarded the dual-branch end-to-end training and pursued fine-tuning only, inspired by the performance improvement presented by (Zoljodi et al., 2024).

### *4.1. Fine-tuning*

To expedite training, the training, testing and validation models were reduced to half of their original sizes in terms of quantity of images processed, and each image was resized to 0.4 of its original height and width, i.e., to 288x512 pixel size. This reduced training time, which was from 1 to 2 hours, to around 5 to 10 minutes per epoch. We also experimented with different types of augmentation, given that cropping out some squares of the input image is a common augmentation step for contrastive learning (Zoljodi et al., 2024; Dippel et al., 2022):

- Basic augmentation: Horizontal Flip (p = 0.5); Color Jitter (Brightness = 0.4, Contrast = 0.4, Saturation = 0.4, Hue = 0.1, p = 0.8); Convert to Gray (p = 0.2); Gaussian Blur (Blur range = from 3 to 7, p = 0.1); Solarize (p = 0.05)
- Augmentation with cropping: same as before, plus Coarse Dropout: 4 random squares of 16x16 pixels would have their values converted to 0 (black)
- Augmentation with random cropping: same as augmentation with cropping, but the number of cropped squares randomly ranges from 1 to 8 and the probability of applying Coarse Dropout is 0.5.

Therefore, the following configurations with the reduced size dataset were obtained:

- FS = Full Supervision with LinkNet
- WS = Weak Supervision pipeline (as presented in Figure 1)
- FT1 = Fine-Tuning the encoders with BYOL optimization using the trained LinkNet as backbone and basic augmentation

- FT2 = Fine-Tuning using augmentation with cropping
- FT3 = Fine-Tuning using augmentation with random cropping

Tables 4, 5 and 6 represent, respectively, the number of epochs before early stopping for each of the configurations, the F1-Scores and IoU results for all configurations per lane marking type. To facilitate comprehension, the improvement or reduction of metric scores are presented in percentages relative to the FS results.

Table 4. Best epoch per training configuration (early stopping after 20 consecutive epochs with no improvement).

| | FS | WS | FT1 | FT2 | FT3 |
|---|---|---|---|---|---|
| Best epoch | 16 | 89 | 29 | 23 | 3 |

Table 5. F1-Score per training configuration.

| F1 Score | FS | WS | FT1 | FT2 | FT3 |
|---|---|---|---|---|---|
| Macro-average | 70.8170 | -22.14% | -0.14% | +1.98% | +2.03% |
| Background | 99.1349 | -0.21% | +0.00% | +0.03% | +0.03% |
| Continuous | 63.2074 | -22.36% | +2.13% | +3.58% | +3.16% |
| Dashed | 63.0277 | -41.38% | -1.57% | +5.34% | +4.63% |
| Unmarked | 79.2775 | -21.89% | +0.01% | +2.88% | +2.87% |

Table 6. IoU score per training configuration.

| IoU | FS | WS | FT1 | FT2 | FT3 |
|---|---|---|---|---|---|
| Macro-average | 58.3512 | -23.29% | -0.28% | +2.42% | +2.49% |
| Background | 98.2866 | -0.41% | +0.01% | +0.05% | +0.05% |
| Continuous | 51.8841 | -27.47% | +3.09% | +4.55% | +4.03% |
| Dashed | 51.6533 | -49.37% | -1.97% | +6.44% | +5.57% |
| Unmarked | 74.9350 | -20.27% | +0.14% | +3.02% | +3.03% |

As can be seen by the results presented in Tables 4 to 6, although the weak supervision pipeline made the training worse overall, using contrastive learning for fine-tuning held positive (albeit modest) results. With a few more epochs that were not showing improvement under the full supervision only training, refinement over all metrics was achieved. Clearly, the BYOL model works better when cropping is added to the augmentation and, since the configurations for FT2 and FT3 were very similar, their results reflect that similarity.

Finally, although FT3 presents higher performance when comparing macro-average values, this is because all FP, FN, TP and TN on the dataset are averaged and the imbalance between classes can cause some distortions. Similarly, the metric values per lane marking type are averaged per batch and also present a slight distortion (values for all monitored metrics are presented on Appendix A). So, we considered that our best performing configuration with the reduced dataset was FT2.

As a final trial, since the improvement was modest, we repeated FT2 configuration with the whole dataset and reduced pixel size. Table 7 presents the results after Full Supervision LinkNet training and fine-tuning.

Table 7. F1-Score and IoU per training configuration with full dataset and reduced pixels.

| | FS - F1 | FS - IoU | FT2 - F1 | FT2 - IoU |
|---|---|---|---|---|
| # of epochs | 12 | 12 | 36 | 36 |
| Macro-average | 73.8627 | 61.4208 | +0.16% | +0.19% |
| Background | 99.1886 | 98.3919 | +0.01% | +0.01% |
| Continuous | 66.7759 | 55.2484 | -1.41% | -1.74% |
| Dashed | 71.3882 | 60.2988 | +0.21% | +0.23% |
| Unmarked | 76.5084 | 71.7966 | -0.67% | -0.72% |

The larger number of inputs for training had a much bigger influence on improving the model's performance than the proposed contrastive fine-tuning, even with our optimal augmentation strategy. That shows that, albeit contrastive learning can achieve modest improvements in sub-optimal conditions, it does not surpass a well-tailored training dataset pre-processing. We still believe that changing training configurations is important to avoid early saturation (our fully supervised model saturated after the 12$^{th}$ epoch), and adding a contrastive learning step to learn embedding features in the latent space can help avoid early saturation. However, in further studies, we advise the pursuit of more alternatives, such as dynamic settling of the learning rate and variate augmentations.

### *4.2. Hyperparameter tuning*

To assess the contribution of the BYOL branch and identify configurations that maximize segmentation performance, an ablation study was conducted on two hyperparameters: loss weighting and learning rate. Monitored metrics and loss values for all trials are available under request.

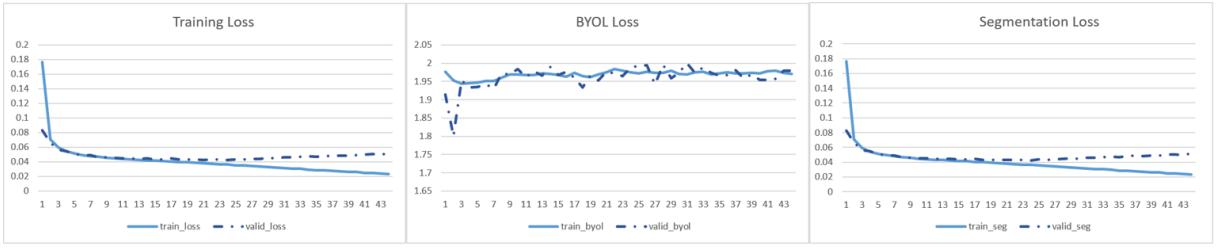
To test loss weights, a new baseline with the best augmentation configuration was run using LinkNet only for training and then balancing different weights for each loss (Cross-Entropy and BYOL) gradually, as presented in Table 8 along with F1-Scores. The test was performed for end-to-end training with each loss, as well as with fine-tuning by reapplying the segmentation-only training or the dual-branch training with the best loss weight configuration. The models were trained until no improvement was recorded after 20 consecutive epochs, and F1-Score reported in the table is the macro-average F1-Score.

Table 8. F1-Score per loss weight configuration.

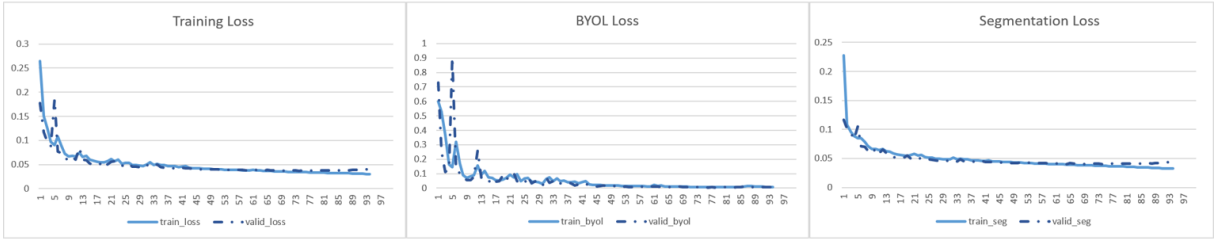
| Model | Loss weight - segmentation | Loss weight - BYOL | Best epoch | F1-Score (%) |
|---|---|---|---|---|
| Baseline | 1.00 | 0.00 | 24 | 74.0900 |
| LR_01 | 0.10 | 0.90 | 79 | -23.13% |
| LR_02 | 0.25 | 0.75 | 95 | -12.40% |
| LR_03 | 0.50 | 0.50 | 99 | -1.96% |
| LR_04 | 0.75 | 0.25 | 95 | -0.49% |
| LR_05 | 0.90 | 0.10 | 74 | +0.76% |
| FT_FS | 1.00 | 0.00 | 2 | +0.11% |
| FT_WS | 0.90 | 0.10 | 4 | +0.71% |

After adapting the augmentation to the best-performing configuration from previous trials, the baseline itself also improved. Regarding the BYOL self-supervised loss, assigning large or equal weights to both losses leads to training dispersion. This is likely because the two losses operate on very different scales: BYOL ranges from [0, 4] while Cross-Entropy ranges from [0, 1]. This is exacerbated by their different starting points during training: while BYOL initializes around 1.0, CE initializes around 0.3, meaning that naively averaging both weights disturbs the training signal.

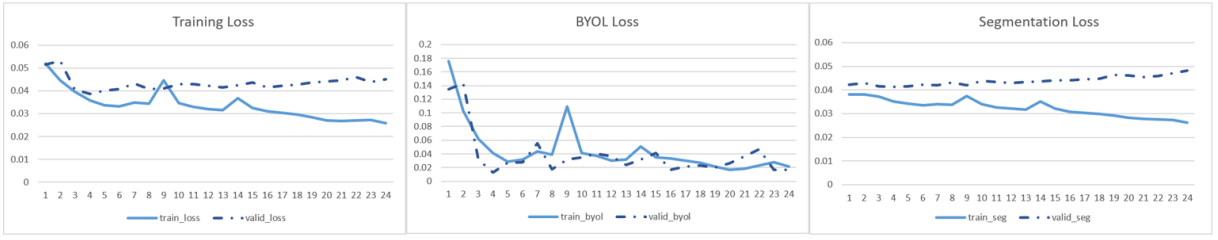
Nevertheless, incorporating the BYOL branch consistently improves performance, albeit modestly. Both end-to-end training and fine-tuning present improvements by approximately 1% when loss weights are of 0.90 for segmentation and 0.10 for BYOL, yielding the best results on the test dataset. Furthermore, we assessed the loss contribution during training: the self-supervised branch accounts for approximately 20% of the total loss in early epochs, and gradually diminishes until it reaches around 2% before early stopping is triggered. Figure 3 illustrates the training logs for three configurations: the baseline, LR_04, and FT_WS.




(a) Baseline




(b) LR_04




(c) FT_WS

Fig. 3 Training logs for three configurations: (a) Baseline; (b) LR_04; and (c) FT_WS.

The decreasing contribution of the BYOL objective throughout training suggests that self-supervision primarily influences the early stages of representation learning. As the segmentation objective becomes increasingly optimized, the model appears to rely progressively less on the auxiliary self-supervised signal. The early saturation allied with significant improvement with the Fine-Tuning configuration also enforces this hypothesis.

Regarding learning rate configuration, both the full supervision (FS) and weak supervision (WS) approaches were evaluated using five learning rates, ranging from 0.1 to 0.00001. For the WS experiments, the loss-weight configuration identified in the previous ablation step was adopted. In addition, the best-performing end-to-end FS configuration was selected for fine-tuning analysis, in which learning rates corresponding to 1, 0.1 and 0.01 times the pre-training learning rate were evaluated. The resulting macro-average F1-Scores for end-to-end training and fine-tuning are presented in Tables 9 and 10, respectively.

Table 9. F1-Score (%) per learning rate for end-to-end training

| Learning rate | FS | WS |
|---|---|---|
| 0.1 | 74.2189 | **74.7961** |
| 0.01 | **74.4998** | 74.2089 |
| 0.001 | 74.0900 | 74.6564 |
| 0.0001 | 73.4237 | 73.2406 |
| 0.00001 | 71.8565 | 71.5094 |

The results indicate that model performance is relatively insensitive to learning rates between 0.1 and 0.001. Within this interval, all configurations achieved macro F1-Scores within approximately 0.7 percentage points of one another, suggesting the existence of a broad performance plateau. Interestingly, the highest scores were obtained at opposite ends of this plateau, with FS achieving its best result at a learning rate of 0.01 and WS at 0.1. The comparable performance observed at both 0.1 and 0.001 suggests that the optimization process is robust across a relatively wide range of learning rates and that, once a sufficiently large learning rate is selected, further tuning yields only marginal gains.

A clear degradation in performance was observed for learning rates below 0.001, with both FS and WS exhibiting lower F1-Scores at 0.0001 and 0.00001. This behaviour indicates that excessively small learning rates limit the effectiveness of optimization within the available training budget. Although WS achieved the highest overall end-to-end performance (74.80%), the difference relative to the best FS configuration remained below one percentage point, suggesting that learning rate selection alone is not the primary factor governing model performance.

Table 10. F1-Score (%) per learning rate for Fine-Tuning

| Learning rate | FT_FS | FT_WS |
|---|---|---|
| 0.01 | 74.6494 | 74.7818 |
| 0.001 | 75.1780 | **75.2594** |
| 0.0001 | 75.2357 | 75.2132 |

The fine-tuning results are overall similar between both training configurations, having achieved an improvement performance of around 1 percentage point with a lower learning rate than the pre-training used. With the same learning rate, the gains did not improve. For 2 out of 3 configuration, the WS configuration outperformed by a timid amount the FS configuration, around 0.1% only. Although the best result overall was found with WS fine tuning and 0.001 learning rate, the differences among runs was not substantial.

Overall, the learning-rate ablation revealed that performance is relatively stable across a broad range of practical learning rates and that fine-tuning provides limited additional benefit. While the dual-branch architecture consistently produced small performance gains over the fully supervised baseline, the magnitude of these gains was comparable to those obtained through conventional hyperparameter optimization. Consequently, the proposed approach should be viewed as a complementary performance-enhancement strategy rather than as a fundamentally superior architecture. Moreover, future work should consider testing multiple approaches for optimization.

### *4.3. Visual results and dataset considerations*

On Figure 4, examples of the worst and best score achieving outputs are given for the fine-tuned model trained with the full dataset, reduced pixels, pre-training learning rate of 0.01 and fine-tuning learning rate of 0.001. The images with index 265 and 103 presented the worst measured F1 and IoU values; as opposed to images index 483 and 430, which had the best F1 and IoU results.

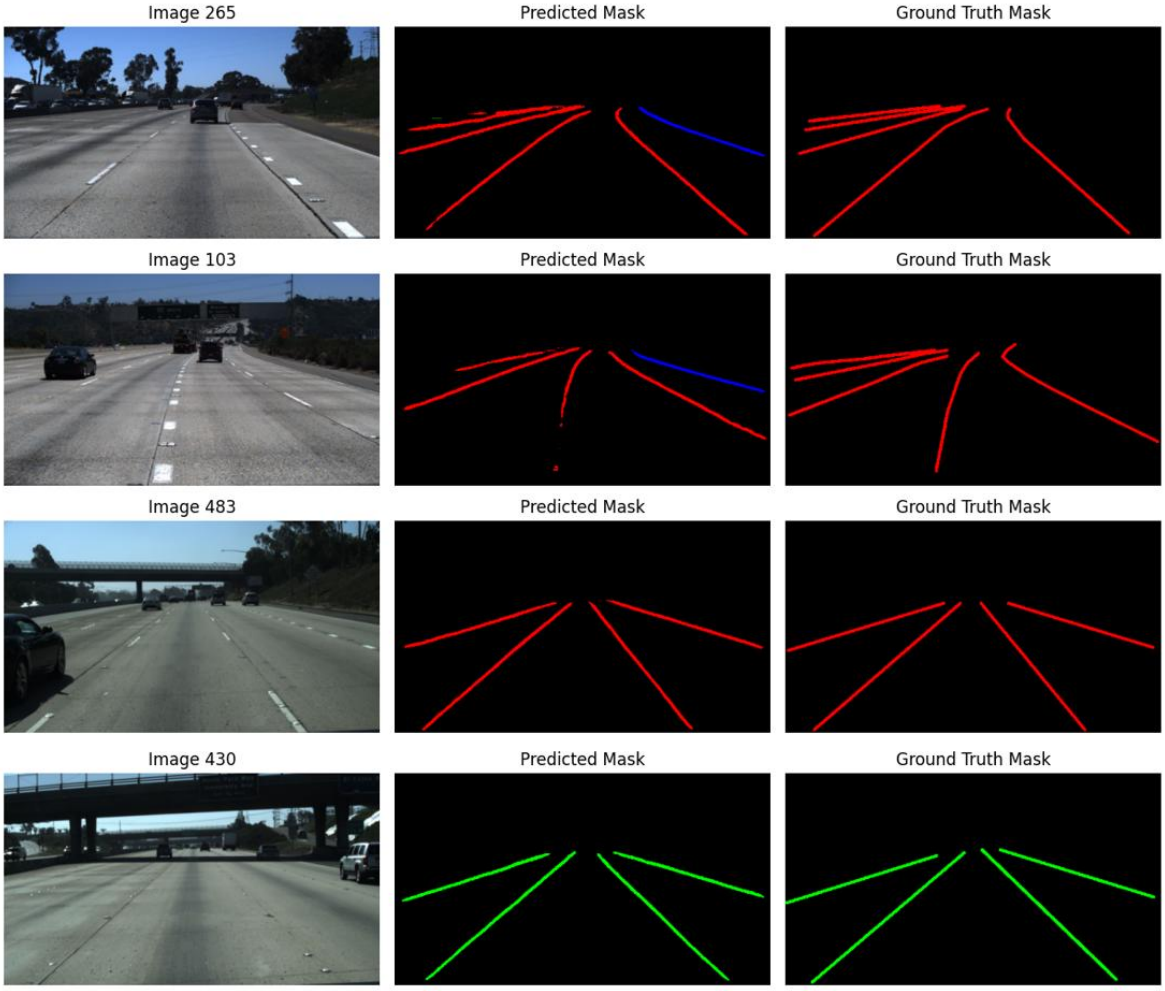



Fig. 4 Best and worst prediction outputs.

It can be seen by Figure 4 that the model can successfully recognize lane marking shapes, and it achieves optimal results when lighting conditions are clear and lane shapes are linear, especially if there are not many different lane class types within the same image, and presents lower but steady performance with curved shapes. Identifying correctly not only the lane type but also its shape would allow to estimate the presence of horizontal curvature in the motorway, which is important information to guide autonomous vehicles and also to estimate risk levels combined with the speed limit and crash amortization attribute presence (Wang et al., 2013; IRAP, 2024).

The worst-performing metrics found on images 265 and 103, however, also reflect difficulties in the labeling process, especially on the presence of multiple lane markings within the same image. On both images, the right-most continuous lane was not included in the original labels, which often does not include all available lanes. Besides that, the left-most lanes are quite far away from the central of the image, making the pixels blurred especially after reducing the image size. This was a difficulty found when relabeling the TuSimple dataset, since the original labels were already inconsistent in situations where multiple lanes were present.

Finally, Figure 5 shows the two images with highest F1 and IoU score differences before and after fine-tuning. Image 74 exhibited the largest decrease in F1 and IoU after fine-tuning, whereas image 723 showed the largest improvement. Each Figure-part is presented in the following order: input image, predicted mask with FS model, predicted mask after FT, ground truth mask.

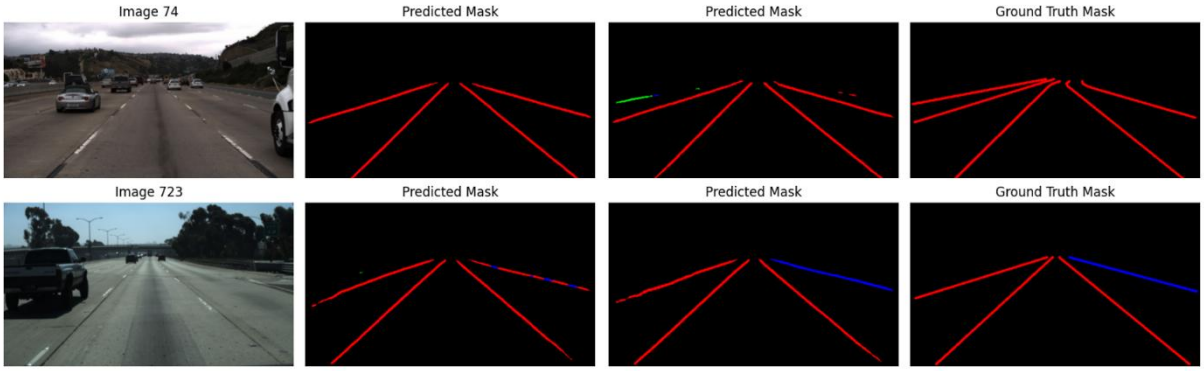



Fig. 5 Visual comparison between LinkNet and Fine-Tuned models.

Even though the metrics score is lower on image 74, the Fine-Tuned model was able to identify an extra lane that had not been detected by the FS model, although the lane marking type assigned was not the same as the labeled one, which reduced the quantitative metrics because the predicted marking class differed from the ground-truth annotation, despite the model identifying additional lane markings that were not detected by the FS model. It was also able to detect some pixels that compose the furthest right lane delineation, which appears to correspond to a paved shoulder delineation marking, and was not present in the original label dataset. As for image 723, the FT model was able to almost perfectly replicate the original labels, while the FS one failed to identify that the right-most lane was continuous, and not dashed. This particular lane has quite difficult detection because there are interventions in the pavement that decrease the markings quality.

These examples suggest that fine-tuning alters the predicted masks in meaningful ways that are not always fully captured by pixel-wise evaluation metrics. That also corroborates how the proposed fine-tuning approach with weak supervision does improve the desired output and provides a more generalizable model by learning embedded representations from the input augmented images.

### *4.4. Comparison with state-of-the-art*

Since the dataset used in this research is novelly configured, a direct comparison with other models cannot be made. Nevertheless, Table 11 shows a comparative analysis between our proposed model and other high-performing reported results for binary lane marking segmentation from the literature.

Table 11. Comparison of metric results between our proposed pipeline and prior work

| Reference | Dataset | Epochs | Lane Marking Type | F1-Score (%) | IoU (%) | Precision (%) | Recall (%) | Accuracy (%) |
|---|---|---|---|---|---|---|---|---|
| Ours (LinkNet with BYOL Fine-Tuning) | TuSimple | 100 | ✓ | 74.74 | 62.44 | 77.73 | 72.40 | 99.20 |
| LinkNet | TuSimple | 100 | ✓ | 74.45 | 62.15 | 77.56 | 72.07 | 99.19 |
| LaneCorrect (Nie et al., 2025) | TuSimple | ? | × | 92.91 | | | | 91.95 |
| CLLD (Zoljodi et al., 2024) | TuSimple | 100 | × | | | | | 94.25 |
| CLLD (Zoljodi et al., 2024) | CuLane | 100 | × | 79.27 | | 88.59 | 71.73 | |

Although the F1-Score value underperforms state-of-the-art methods on the TuSimple benchmark (Nie et al., 2025), it is comparable to values presented for the more challenging CULane dataset (Nie et al., 2025; Zoljodi et al., 2024). This outcome is encouraging, as the introduction of multiclass labels in our setting inherently increases task difficulty compared to binary lane detection. Moreover, the precision and recall values are reasonably aligned, suggesting that the model maintains a balanced behavior between correctly identifying and classifying lane markings and avoiding false detections, without one metric disproportionately compensating for the other. Regarding the

accuracy result, it is not an appropriate metric to be monitored because of the imbalance of the data that results in a large quantity of True Negative pixels regardless of model performance, and was mainly included in Table 9 for completeness, as it has been often reported by other researchers. It can be seen that it has consistent high results, over 90%, even though some of the other metrics change drastically.

It is also worth noting that the backbone of the proposed approach is a generic segmentation network, which was not originally designed for lane detection. Replacing it with a model tailored for this task, such as CLRNet (Zheng et al., 2022), could further enhance performance, as demonstrated by Zoljodi et al. (2024). Also, our fine-tuned model has achieved the same performance as the fully supervised LinkNet model using a reduced size dataset, that made the model training much lighter and less computationally expensive.

Finally, our results might be understating the actual performance of the model given the presence of noise and possibly wrongfully labelled lanes on our ground truth masks. On future work, we plan on assessing the lowest performing input images to correct mislabelled ground truth lanes and build a more solid, reliable dataset for multiclass lane marking segmentation.

## 5. Conclusions, Limitations and Future Work

In this work, we have trained a dual-branch weakly supervised model for lane marking segmentation for three type classes: continuous, dashed and unmarked. Our model was composed of a self-supervised BYOL branch, trained to understand intrinsic relations from two differently augmented versions of the input image, and of a LinkNet segmentation branch, a high performing and lightweight segmentation model, trained with full supervision. Our dataset was derived from open source TuSimple dataset, and lane types were manually assigned after the original lane labels were converted to individual entities based on their graph-like structure extracted with NetworkX library. The train, validation and test division suggested by TuSimple was not followed in order to assure equal distribution between lane marking types for all partitions.

In short, the main contribution of this work lies in the integration of contrastive learning, weak supervision and multiclass lane segmentation into a unified framework, demonstrating effective results. Although the weakly supervised training did not yield exceptional results, fine-tuning the LinkNet model with BYOL contrastive learning did achieve promising results, especially when training with a reduced dataset. Both F1-Score and IoU were lower than TuSimple benchmark but comparable to other more challenging datasets, which was expected due to the increase of difficulty occasioned by the multiclass task assignment. Precision and recall achieved similar values, indicating that the output is quite balanced. The presence of occlusions and noise in the labels were the main factors that pulled the metrics down.

Some limitations are present in our work. Ablation tests didn't include, for example, changing the dataset size or use of pre-trained weights for out backbone. Evaluating the model on different datasets would also shed light on its transferability. For future work, we propose integrating the outputs of the proposed model with other transportation-related tasks. Also, although consistent trends favouring the weakly supervised configuration were observed, the absolute improvements remained below one percentage point. Future work should evaluate multiple training seeds to quantify the statistical significance and reproducibility of these gains. The predictions could be used to guide object detection algorithms by providing additional semantic context or serve as complementary attribute data to enhance road safety analysis and infrastructure management.

## Acknowledgements

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101119590.

## References

Chaurasia, A., Culurciello, E, 2017. LinkNet: Exploiting Encoder Representations for Efficient Semantic Segmentation. 2017 IEEE Visual Communications and Image Processing (VCIP), 1–4. https://doi.org/10.1109/VCIP.2017.8305148

Chen, T., Kornblith, S., Norouzi, M., Hinton, G., 2020. A simple framework for contrastive learning of visual representations. In Proceedings of the 37th International Conference on Machine Learning (Vol. 119, pp. 1597–1607). PMLR.

Chiu, K. Y., & Lin, S. F., 2005. Lane detection using color-based segmentation. In 2005 IEEE Intelligent Vehicles Symposium (pp. 706–711). https://doi.org/10.1109/IVS.2005.1505186

Dippel, J., Lenga, M., Goerttler, T., Obermayer, K., Höhne, J., 2022. Transfer Learning for Segmentation Problems: Choose the Right Encoder and Skip the Decoder (No. arXiv:2207.14508). arXiv. https://doi.org/10.48550/arXiv.2207.14508

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. Á., Guo, Z. D., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., Valko, M., 2020. Bootstrap your own latent: A new approach to self-supervised learning. In Advances in Neural Information Processing Systems 33 (NeurIPS 2020).

He, K., Fan, H., Wu, Y., Xie, S., Girshick, R., 2020. Momentum contrast for unsupervised visual representation learning. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 9726-9735). IEEE. https://doi.org/10.1109/CVPR42600.2020.00975

Hoang, T., Hong, H., Vokhidov, H., Park, K., 2016. Road Lane Detection by Discriminating Dashed and Solid Road Lanes Using a Visible Light Camera Sensor. Sensors 16, 1313. https://doi.org/10.3390/s16081313

Iakubovskii, P., 2019. Segmentation Models Pytorch. GitHub Repos.

International Road Assessment Programme – iRAP, 2024. iRAP Coding Manual Version 5.4 – Drive on Right Edition (English). Available in: https://irap.org/specifications/. Access on 29th of October of 2025.

Khan, M. A. H., Ganeriwala, P., Lehman, S. M., Bhattacharyya, S., Alvarez, A., Neogi, N., 2025. Adapt, But Don't Forget: Fine-Tuning and Contrastive Routing for Lane Detection under Distribution Shift. [Preprint]. arXiv. https://doi.org/10.48550/arXiv.2507.18653

Löwens, C., Thyssens, D., Andersson, E., Jenkins, C., Schmidt-Thieme, L., 2023. DeepStay: Stay Region Extraction from Location Trajectories using Weak Supervision. 10.48550/arXiv.2306.06068.

Mamun, A. A., Ping, E,. P., Hossen, J., Tahabilder, A., Jahan, B., 2022. A Comprehensive Review on Lane Marking Detection Using Deep Neural Networks. In: Sensors, Volume 22, Issue 19, 7682. https://doi.org/ 10.3390/s22197682

Meletis, P., Romijnders, R., Dubbelman, G., 2019. Data selection for training semantic segmentation CNNs with cross-dataset weak supervision. In 2019 IEEE Intelligent Transportation Systems Conference (ITSC) (pp. 3682-3688). https://arxiv.org/pdf/1907.07023

Merugu, K., Adarsh, S., 2022. Multi lane detection, curve fitting and lane type classification, in: 2022 IEEE 19th India Council International Conference (INDICON). Presented at the 2022 IEEE 19th India Council International Conference (INDICON), IEEE, Kochi, India, pp. 1–6. https://doi.org/10.1109/INDICON56171.2022.10039722

Muthalagu, R., Bolimera, A., Kalaichelvi, V., 2021. Vehicle lane markings segmentation and keypoint determination using deep convolutional neural networks. Multimedia Tools and Applications, *80*(8), 11201–11215. https://doi.org/10.1007/s11042-020-10248-2

Nie, M., Cai, X., Xu, H., Zhang, L., 2025. LaneCorrect: Self-Supervised Lane Detection. Int. J. Comput. Vis. 133, 4894–4908. https://doi.org/10.1007/s11263-025-02417-3

Porto, J.A., Ziakopoulos, A., Yannis, G., 2025. Road segmentation made simple: a practical comparison of segmentation models and post-processing techniques. In: 12th International Conference on Transportation Research (ICTR 2025). Available in: https://www.nrso.ntua.gr/geyannis/wp-content/uploads/geyannis-pc603.pdf

Tran, L. A., Le, M. H., 2019. Robust U-Net-based road lane markings detection for autonomous driving. In 2019 International Conference on System Science and Engineering (ICSSE) (pp. 62–66). https://doi.org/10.1109/ICSSE.2019.8823532

Wang, C., Quddus, M. A., Ison, S. G., 2013. The effect of traffic and road characteristics on road safety: A review and future research direction. In: Safety Science, Volume 57, pp. 264-275. https://doi.org/10.1016/j.ssci.2013.02.012

Yoo, S., Lee, H., Myeong, H., Yun, S., Park, H., Cho, J., Kim, D., 2020. End-to-end lane marker detection via row-wise classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (pp. 4335–4343). https://doi.org/10.1109/CVPRW50498.2020.00511

Zakaria, N. J., Shapiai, M. I., Ghani, R. A., Yassin, M. N. M., Ibrahim, M. Z., Wahid, N., 2023. Lane detection in autonomous vehicles: A systematic review. IEEE Access, *11*, 3729–3765. https://doi.org/10.1109/ACCESS.2023.3234442

Zheng, T., Huang, Y., Liu, Y., Tang, W., Yang, Z., Cai, D., He, X., 2022. CLRNet: Cross Layer Refinement Network for Lane Detection, in: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Presented at the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, New Orleans, LA, USA, pp. 888–897. https://doi.org/10.1109/CVPR52688.2022.00097

Zoljodi, A., Abadijou, S., Alibeigi, M., Daneshtalab, M., 2024. Contrastive Learning for Lane Detection via cross-similarity. Pattern Recognit. Lett. 185, 175–183. https://doi.org/10.1016/j.patrec.2024.08.007

## Appendix A – Monitored metrics results per ablation trial

Table A.1 – Metric results for LinkNet model with reduced dataset

| Line Type | F1 Score | IoU | Precision | Recall | Accuracy |
|---|---|---|---|---|---|
| Average | 70.8170 | 58.3512 | 76.7628 | 66.9648 | 99.1086 |
| Background | 99.1349 | 98.2866 | 98.8973 | 99.3745 | 98.3148 |
| Continuous | 63.2074 | 51.8841 | 71.1930 | 68.0855 | 99.4556 |
| Dashed | 63.0277 | 51.6533 | 68.7197 | 68.7938 | 99.1033 |
| Unmarked | 79.2775 | 74.9350 | 86.8190 | 81.8989 | 99.5645 |

Table A.2 – Metric results for Weak Supervision model with reduced dataset

| Line Type | F1 Score | IoU | Precision | Recall | Accuracy |
|---|---|---|---|---|---|
| Average | 55.1369 | 44.7615 | 75.9441 | 48.9048 | 98.9134 |
| Background | 98.9289 | 97.8825 | 98.2488 | 99.6208 | 97.9030 |
| Continuous | 49.0746 | 37.6333 | 62.7102 | 55.9230 | 99.3616 |
| Dashed | 36.9477 | 26.1530 | 52.4933 | 53.4842 | 98.9311 |
| Unmarked | 61.9201 | 59.7422 | 86.0619 | 70.1078 | 99.4561 |

Table A.3 – Metric results for Fine Tuning 1 (FT1) configuration

| Line Type | F1 Score | IoU | Precision | Recall | Accuracy |
|---|---|---|---|---|---|
| Average | 70.7209 | 58.1905 | 76.0773 | 67.1474 | 99.1040 |
| Background | 99.1397 | 98.2960 | 98.9189 | 99.3623 | 98.3243 |
| Continuous | 64.5517 | 53.4886 | 73.1442 | 67.7807 | 99.4606 |
| Dashed | 62.0373 | 50.6352 | 66.5883 | 70.1266 | 99.0970 |
| Unmarked | 79.2857 | 75.0427 | 86.6243 | 81.8107 | 99.5632 |

Table A.4 – Metric results for Fine Tuning 2 (FT2) configuration

| Line Type | F1 Score | IoU | Precision | Recall | Accuracy |
|---|---|---|---|---|---|
| Average | 72.2163 | 59.7629 | 77.1169 | 68.9242 | 99.1370 |
| Background | 99.1621 | 98.3400 | 98.9594 | 99.3664 | 98.3685 |
| Continuous | 65.4733 | 54.2435 | 72.5417 | 70.0986 | 99.4774 |
| Dashed | 66.3951 | 54.9822 | 70.9654 | 71.0258 | 99.1399 |
| Unmarked | 81.5620 | 77.1970 | 88.3657 | 82.6214 | 99.5741 |

Table A.5 – Metric results for Fine Tuning 3 (FT3) configuration

| Line Type | F1 Score | IoU | Precision | Recall | Accuracy |
|---|---|---|---|---|---|
| Average | 72.2527 | 59.8037 | 77.0404 | 69.0513 | 99.1365 |
| Background | 99.1620 | 98.3397 | 98.9652 | 99.3602 | 98.3683 |
| Continuous | 65.2029 | 53.9731 | 72.1446 | 70.2517 | 99.4769 |
| Dashed | 65.9476 | 54.5329 | 70.3440 | 71.1632 | 99.1377 |
| Unmarked | 81.5536 | 77.2031 | 88.3539 | 82.6502 | 99.5749 |

Table A.6 – Metric results for LinkNet with full dataset and reduced image pixel-size

| Line Type | F1 Score | IoU | Precision | Recall | Accuracy |
|---|---|---|---|---|---|
| Average | 73.8627 | 61.4208 | 77.3814 | 71.1702 | 99.1740 |
| Background | 99.1886 | 98.3919 | 99.0126 | 99.3658 | 98.4196 |
| Continuous | 66.7759 | 55.2484 | 72.7639 | 71.0890 | 99.4863 |
| Dashed | 71.3882 | 60.2988 | 76.1628 | 71.8399 | 99.1949 |
| Unmarked | 76.5084 | 71.7966 | 80.5929 | 84.4830 | 99.5954 |

Table A.7 – Metric results for Fine-Tuned LinkNet with full dataset and reduced image pixel-size

| Line Type | F1 Score | IoU | Precision | Recall | Accuracy |
|---|---|---|---|---|---|
| Average | 73.9844 | 61.5376 | 77.7740 | 71.0894 | 99.1799 |
| Background | 99.1947 | 98.4039 | 99.0083 | 99.3824 | 98.4314 |
| Continuous | 65.8353 | 54.2858 | 72.2679 | 70.6778 | 99.4868 |
| Dashed | 71.5410 | 60.4374 | 76.7331 | 71.6009 | 99.2003 |
| Unmarked | 75.9924 | 71.2822 | 80.0406 | 84.6449 | 99.6009 |