Real-Time Anomaly Segmentation for Road Scenes

Emanuele Sanna
Politecnico di Torino
s330417@studenti.polito.it

Alessandro Manera
Politecnico di Torino
s324246@studenti.polito.it

Giuseppe Monteasi
Politecnico di Torino
s319203@studenti.polito.it

Abstract

This study addresses real-time anomaly segmentation in road scenes, a critical task for applications like autonomous driving. We evaluate baseline segmentation models (ENet, ERFNet, BiSeNet) pretrained on Cityscapes, with different metrics and methods, incorporating enhancements such as temperature scaling, void classifiers, and pruning and quantization functions. Performance is assessed on benchmark datasets using AuPRC, FPR95, and mIoU metrics. Results highlight the robustness of MaxLogit, the effectiveness of BiSeNet, and the benefits of pruning and quantization in terms of memory and time savings for anomaly segmentation. These findings offer insights into building efficient, reliable systems for real-world environments. The source code of this project is available at https://github.com/emariesanna/AnomalySegmentation-AMLProject.

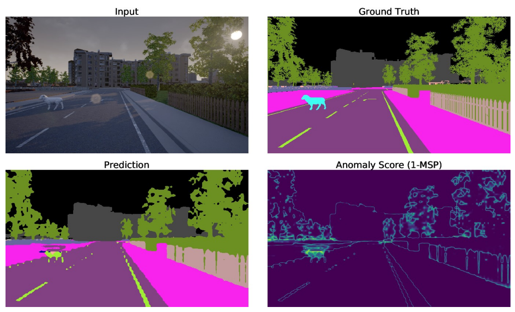

Figure 1. An example image for anomaly segmentation in road scenes.

1. Introduction

Anomaly segmentation is a critical task in computer vision that involves identifying regions within an image that deviate from expected patterns. This capability has significant real-world applications, including detecting road obstacles for autonomous driving vehicles or identifying defective objects in industrial systems. Anomalies often represent unpredictable or rare events, such as fallen debris on a roadway, making their detection essential for safety and operational efficiency. In this context, per-pixel anomaly segmentation focuses on the identification of anomalous regions at the pixel level. This task is particularly challenging due to the need to distinguish between In-Distribution (ID) and Out-of-Distribution (OOD) samples that the model has not encountered during training. In the domain of autonomous driving, per-pixel anomaly segmentation ensures that the system can accurately localize and respond to hazards, even when faced with novel or unexpected scenarios. Achieving real-time performance for per-pixel anomaly segmentation is vital for its deployment in safety-critical applications, such as autonomous vehicles, which must process sensor data with minimal latency to make fast and reliable decisions. This requires methods that strike a balance between computational efficiency and high detection accuracy. In this paper, we explore per-pixel anomaly segmentation through a series of experiments designed to evaluate and enhance its performance. We begin by establishing and testing three baseline models for image segmentation (ENet [13], ERFNet [15], and BiSeNet [17]), pretrained on Cityscapes [4] and tested on different task-specific datasets with different methods, followed by the application of temperature scaling for confidence calibration. Additionally, we introduce a Void Classifier [5] to explicitly leverage OOD knowledge. Furthermore, we analyze the effects of pruning and quantization techniques to reduce latency, thereby accelerating the anomaly detection process, albeit at the cost of some accuracy.

2. Related Works

Real-time semantic segmentation requires a trade-off between computational efficiency and the ability to capture spatial and contextual information. This section reviews three key architectures, which we used, designed for this purpose: ENet, ERFNet, and BiSeNet.

ENet (Efficient Neural Network): ENet [13] is a lightweight encoder-decoder model optimized for resource-limited environments. The encoder compresses spatial information early using downsampling and bottleneck modules, which reduce feature dimensionality via 1×1 projections followed by regular, dilated, or asymmetric convolutions (Figure 2). The minimalistic decoder focuses on upsampling the encoded representations for pixel-level predictions. By limiting decoder complexity, ENet achieves high computational efficiency, making it well-suited for embedded systems.

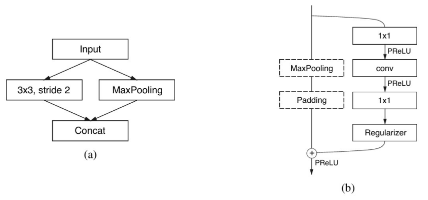

Figure 2. (a) ENet initial block: MaxPooling with non-overlapping 2×2 windows; convolution with 13 filters, resulting in 16 feature maps. (b) ENet bottleneck module: *conv* can be regular, dilated, full convolution (deconvolution), or decomposed asymmetric convolution.

ERFNet (Efficient Residual Factorized Network): ERFNet [15] extends ENet's efficiency by introducing non-bottleneck-1D modules (Figure 3), which replace traditional 2D convolutions with factorized 1D convolutions, significantly reducing computational cost while preserving representational power. The encoder utilizes these modules with downsampling and dilated convolutions for multi-scale feature extraction, while the lightweight decoder employs transposed convolutions to recover spatial resolution. This balance between speed and accuracy makes ERFNet well-suited for real-time applications.

BiSeNet (Bilateral Segmentation Network): BiSeNet [17] employs a dual-path structure to address the trade-off between spatial detail preservation and contextual understanding (Figure 4). The Spatial Path (SP) maintains high-resolution features using three convolutional layers with a stride of 2, while the Context Path (CP), leveraging lightweight backbones like Xception, extracts global features through aggressive downsampling and global average pooling. A Feature Fusion Module (FFM) merges SP and CP outputs, while an Attention Refinement Module (ARM)

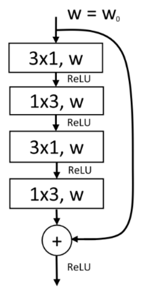

Figure 3. ERFNet non-bottleneck-1D module.

enhances feature relevance. This design ensures efficient segmentation with strong accuracy.

This section highlights key contributions of these architectures, demonstrating various strategies for balancing efficiency and performance in real-time segmentation.

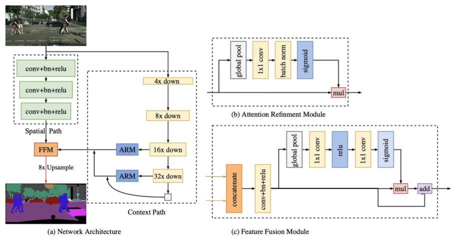

Figure 4. BiSeNet's dual-path architecture: Spatial Path (SP), Context Path (CP), and Feature Fusion Module (FFM).

3. Methods

Anomaly detection in semantic segmentation relies on leveraging the model's predictive confidence and internal feature representations. This section presents four primary techniques: Maximum Softmax Probability (MSP), Maximum Logit (MaxLogit), Maximum Entropy (MaxEntropy), and the Void Classifier. Each method assigns a pixel-wise anomaly score $s(x) \in \mathbb{R}^{|Z|}$, where $x \in X$ denotes an image, with higher scores indicating an anomalous region. The variable Z represents the set of pixel coordinates, and X comprises a dataset of N images. The softmax function is denoted as $\sigma(\cdot)$, while $f_z^c(x)$ represents the logit for class c at pixel z.

3.1. Maximum Softmax Probability (MSP)

MSP [8] quantifies model confidence by computing the highest softmax probability per pixel. The anomaly score is formulated as:

$$s_z(x) = 1 - \max_{c \in C} \sigma(f_z^c(x)). \tag{1}$$

A low maximum probability signifies an increased likelihood of an anomaly. However, MSP is susceptible to overconfidence in softmax outputs, potentially leading to miscalibrated uncertainty estimates. To mitigate this, *temperature scaling* [1] [6] is introduced as a post-processing calibration mechanism. The refined anomaly score is expressed as:

$$s_z(x) = 1 - \max_{c \in C} \sigma(f_z^c(x)/T), \tag{2}$$

where T represents the temperature parameter, which smooths the probability distribution and enhances robustness.

3.2. Maximum Logit (MaxLogit)

MaxLogit [7] bypasses the softmax transformation, operating directly on logits to mitigate the issue of artificially inflated confidence scores. The anomaly score is defined as:

$$s_z(x) = -\max_{c \in C} f_z^c(x). \tag{3}$$

By leveraging raw logits, this method provides a more stable and computationally efficient measure of anomaly likelihood.

3.3. Maximum Entropy (MaxEntropy)

MaxEntropy [3] quantifies uncertainty by computing the Shannon entropy of softmax outputs:

$$s_z(x) = -\sum_{c \in C} \sigma(f_z^c(x)) \log(\sigma(f_z^c(x))). \tag{4}$$

Higher entropy values correspond to increased model uncertainty, making this approach well-suited for detecting out-of-distribution (OOD) pixels. Compared to MSP, MaxEntropy offers a more refined uncertainty estimate, though at the cost of increased computational complexity.

3.4. Void Classifier

This approach incorporates an explicit ”void” class into the segmentation network, allowing the model to directly classify anomalies. Formally, given a network $f : X \to \mathbb{R}^{|Z| \times (|C|+1)}$, the anomaly score at each pixel is computed as:

$$s_z(x) = \sigma(f_z^{void}(x)), \quad x \in X. \tag{5}$$

By explicitly modeling anomalies, the void classifier reduces reliance on post-hoc anomaly detection. However, its performance is contingent on the availability of accurately annotated void regions during training.

[1]https : / / github . com / gpleiss / temperature _ scaling?tab=readme-ov-file

3.5. Pruning

Pruning enhances model efficiency by systematically removing less significant parameters, thereby reducing computational overhead. In this work, we applied **structured L_2-norm pruning** to ERFNet’s convolutional layers, eliminating 30% of the least significant filters along the output channel dimension. This technique optimally balances model compression with representational capacity, ensuring minimal loss of critical features [12].

3.6. Quantization

Quantization reduces numerical precision from floating-point to fixed-point representations, significantly decreasing memory requirements and computational latency. We employed PyTorch’s FX Graph Mode **Post-Training Static Quantization** (PTQ), which automates the quantization process via symbolic tracing [10, 18]. The workflow consists of:

- **Model Preparation**: Initializing the pre-trained ERFNet model in evaluation mode.
- **Quantization Configuration**: Defining quantization parameters using PyTorch’s standard settings.
- **Instrumentation**: Inserting observers via `prepare_fx()` to gather activation statistics.
- **Calibration**: Running inference on a representative dataset to compute quantization parameters.
- **Conversion**: Transforming the model into its quantized form using `convert_fx()`.

This process effectively reduces computational demands while preserving segmentation accuracy.

3.7. Distillation

Distillation is a technique that transfers knowledge from a larger, more complex model (the teacher) to a smaller, more efficient model (the student) [9]. In this context, we apply distillation to enhance the performance of our segmentation models by leveraging the soft output distributions of a pre-trained teacher model. Instead of training the student solely on hard labels (i.e., one-hot encoded ground truth), the student is guided to mimic the teacher’s predictions, which often contain rich information about inter-class relationships and uncertainty. This is achieved by minimizing the *Kullback-Leibler (KL) divergence* [16] between the output probability distributions (typically the softmax outputs) of the teacher P and the student Q. KL divergence is a statistical measure of how one probability distribution diverges from a second, expected distribution—in this case, how the student's predictions deviate from the teacher's. It is defined as:

$$KL(P \parallel Q) = \sum_i P(i) \log \left(\frac{P(i)}{Q(i)} \right) \tag{6}$$

where $P(i)$ is the probability of class i predicted by the teacher, and $Q(i)$ is the probability of class i predicted by the student. This formulation encourages the student to produce similar confidence scores across all classes, not just the correct one, effectively smoothing the learning process. This approach allows the student model to learn nuanced patterns from the teacher, leading to better generalization even with fewer parameters. As a result, distillation enables us to retain much of the teacher's performance while maintaining the computational efficiency required for real-time anomaly segmentation applications.

4. Experiments

In our experiments, we evaluate an anomaly segmentation framework using inference methods like Maximum Softmax Probability (MSP), Maximum Logit (MaxLogit), and Maximum Entropy (MaxEntropy). We also investigate the impact of Temperature Scaling for calibration and the use of the void class from the Cityscapes dataset for anomaly detection. Additionally, we explore pruning and quantization techniques to improve execution speed and efficiency. Experiments are conducted on benchmark datasets such as RoadAnomaly, Fishyscapes, and SegmentMelfYouCan, using pre-trained models like ENet, ERFNet, and BiSeNetV1.

4.1. Metrics

We evaluate performance using three metrics: Area under the Precision-Recall Curve (AuPRC), False Positive Rate at 95% True Positive Rate (FPR95), and Mean Intersection over Union (mIoU). The **AuPRC** metric quantifies how well the anomaly scores separate anomalies from non-anomalies. The metric is threshold-independent, meaning it considers how precision and recall change as the threshold for classifying points as anomalies is varied. A higher AuPRC indicates better separation between anomalies and non-anomalies, with a value closer to 1 being ideal (It is worth being mentioned that in our tables we reported the AuPRC values in percentage). The **FPR95** metric evaluates the false positive rate when the true positive rate is 95%. The positive class is defined as "anomaly", and false positives are non-anomaly pixels incorrectly predicted as anomalies. The **mIoU** metric instead quantifies the overlap between predicted and ground truth segmentation masks. It is defined as:

$$mIoU = \frac{1}{C} \sum_{c=1}^{C} \frac{TP_c}{TP_c + FP_c + FN_c}, \tag{7}$$

where C is the number of classes, and TP_c, FP_c, and FN_c are the true positive, false positive, and false negative pixels for class c, respectively. The mIoU values reported in the tables of this paper (1, 2, 3, 4) refer to a mIoU evaluation over the Cityscapes dataset [4]. Furthermore, to measure the effectiveness of pruning and quantization on ERFNet, we also considered FLOPS (Floating Point Operations), number of trainable model parameters and anomaly segmentation performance.

4.2. Datasets and Benchmarks

Here, we describe the main datasets and benchmarks relevant to our work. In our experiments we have only used the validation splits of these benchmark datasets, which are publicly available for download but contain a limited number of different road surfaces and diverse obstacle types than the whole dataset benchmarks.

Cityscapes [4] is a widely-used dataset for semantic segmentation in urban driving scenarios. It consists of 5,000 high-resolution images with dense pixel-level annotations across 19 semantic classes, captured in diverse European cities. Cityscapes provides a strong foundation for segmentation models, particularly for road scene understanding.

Fishyscapes [1] is a benchmark designed to evaluate anomaly detection in semantic segmentation. It includes three datasets, but only two were used in our work: FS Static and FS Lost and Found. *FS Static* is based on the Cityscapes validation set and is divided into a public validation set of 30 images with 30 OOD objects overlaid, and a hidden test set of 1000 images. *FS Lost and Found* is derived from the Lost and Found dataset [14] and consists of 100 validation images and 275 test images with pixel-level annotations of small, anomalous objects on the road.

RoadAnomaly [11] focuses on detecting anomalies in real-world road scenes. This dataset features 60 images with pixel-level annotations with various anomalous objects, such as animals or atypical vehicles, appearing in unpredictable locations within the image, making it a challenging test for anomaly segmentation models.

SegmentMeIfYouCan [2] is a benchmark for anomaly segmentation that introduces two key datasets: RoadAnomaly21 and RoadObstacle21. *RoadAnomaly21* contains 100 real-world test images and 10 validation images where anomalies can appear anywhere, emphasizing general anomaly detection. *RoadObstacle21* consists of 327 test images and 30 validation images, restricting the region of interest to the drivable road area, and focuses on the detection of potential hazards such as fallen objects, with annotations tailored for obstacle segmentation tasks. In both datasets, the pixel-level annotations include three classes: anomaly/obstacle, not anomaly/not obstacle, void.

4.3. Implementation details

In this section, we detail our implementation protocol for each model. Our first plan involved exploiting the pre-trained versions of ENet [2], ERFNet [3] and BiSeNetV1 [4] available in their official GitHub repositories. Unfortunately we were not able to get meaningful results using the pre-trained versions of ENet we found online. We then decided to train the model from scratch. We ended up training it for 560 epochs, but we decided to show also the results of ENet at 340 epochs as an intermediate snapshot in Table 3. All models were trained or pre-trained on the Cityscapes dataset with 19 semantic classes. For the void classifier experiment (Section 4.6), we fine-tuned the models for 30 epochs to include the void class in the anomaly score by freezing all layers except the final one. For pruning the ERFNet model (Section 4.7), we first focused on the Conv-2D layers, zeroing 30% of the weights and selecting which weights to keep based on the L_2-norm technique. Then, we applied a brief fine-tuning phase (10 epochs) to adjust the remaining weights. Next, for quantization, we applied observers to the weights. After that, we ran one evaluation epoch to calibrate the weights and calculate the zero-point and scale — the two values that allow us to discretize the weights.
We pre-processed the data by resizing each image to 512×1024 pixels for the fine-tuning, while the training of ENet was made on 1024×2048. The data augmentations and hyperparameters were adopted directly from the original papers to ensure consistency with the authors' implementations. Below, we detail the specific settings and configurations used for each model. For all the models, we used the Adam optimizer with an initial learning rate of 5×10^{-5} and a weight decay of 10^{-4} (for the training of ENet we actually set the initial learning rate to 0.04, choosing a value this high because we needed to decrease the loss faster since we started training the model from scratch). The learning rate was adjusted using a LambdaLR scheduler based on Equation 8. The Lambda LR scheduler is defined as:

$$\lambda(t) = \left(1 - \frac{t-1}{T}\right)^{0.9}, \tag{8}$$

where t is the epoch and T is the total number of epochs.

4.4. Baselines

To compare the effect of different methods (MSP, MaxLogit, MaxEntropy) for anomaly segmentation, we evaluated the performance of an ERFNet model pre-trained with 19 Cityscapes classes on different datasets: RoadAnomaly21, RoadObstacle21, Fishyscapes Static Lost and Found, Fishyscapes Static, RoadAnomaly. The performance results reported in Table 1 show that the MaxLogit method generally outperforms all other methods, because it can effectively distinguish between classes that are very similar. On the contrary, the MSP method, which relies on Softmax, tends to reduce the logits, making the differences between classes less pronounced and harder to differentiate.

4.5. Temperature Scaling

In this experiment, we implemented Temperature Scaling in the MSP method using Equation 2 and we studied the effect of different temperature values on benchmark datasets. We tried different temperatures in a range between 0.5 and 1.1. To find the best temperature we then trained the temperature parameter as the temperature is a learnable parameter [5]. The results reported in Table 2 show that MSP with a temperature of 2.15 outperforms all other temperature values, indicating that the network requires calibration.

4.6. Void Classifier

In this experiment, we fine-tuned ERFNet, ENet and BiSeNetV1 by enabling the void output channel, representing the 20th class in the Cityscapes dataset, typically used for background or unannotated areas. We reinterpreted it as an anomaly class, encompassing all elements outside of the 19 predefined Cityscapes categories. Once again, while ERFNet and BiSeNet were pre-trained on Cityscapes, we had to first train ENet ourselves. Then, to adapt them, we fine-tuned for 30 epochs on the Cityscapes dataset setting the weight of the void class to 1, in order to approximate an anomaly distribution using the dataset's void regions directly during training, obtaining the *Void Classifier* [2] defined in Section 3.4. During inference, anomaly detection was performed by isolating the void output class and treating it as an anomaly score. Table 3 presents the performance of networks trained as Void Classifiers, tested on various benchmark datasets. BiSeNet consistently achieves the highest AuPRC scores across all datasets, notably outperforming others on SMIYC RA-21 (46.79%) and FS Static (42.52%), demonstrating superior precision and recall in detecting void classes. On the other hand, ERFNet achieves the best FPR95 results on most datasets, excelling in FS L&F (13.17%), reflecting its robustness in minimizing false positives. Comparing ERFNet trained as a Void Classifier (first row of Table 3) with baseline methods (Table 1), the Void Classifier generally performs worse, except for improvements in FS L&F (FPR95) and FS Static (AuPRC). Also the mIoU is slightly lower, suggesting that the model may benefit from further fine-tuning.

[2]https : / / github . com / yassouali / pytorch - segmentation
[3]https://github.com/Eromera/erfnet_pytorch
[4]https://github.com/CoinCheung/BiSeNet

[5]https://arxiv.org/abs/1706.04599

4.7. Effects of Pruning and Quantization

The choice of L_2-norm structured pruning was driven by its ability to remove entire filters, simplifying the model and reducing computational load more effectively than unstructured pruning [12]. For quantization, we employed static quantization [10], which is well-suited for models with many linear layers and allows for runtime improvements via integer arithmetic. As shown in Table 4, quantizing ERFNet reduced mIoU from 64.31% to 44.80%, with relatively stable AuPRC and FPR95 across datasets. The quantized model also achieved a significantly faster inference time (1.62 fps) compared to the original (0.22 fps) and pruned (0.67 fps) versions (Table 5). This confirms quantization's benefits for latency and model compactness, despite reduced segmentation accuracy. Combining pruning and quantization further improved inference speed (1.76 fps) but decreased mIoU to 41.06%. Nonetheless, this version achieved the best FPR95 on SMIYC RO-21 (95.31%) and Road Anomaly (89.39%), making it viable where speed outweighs accuracy. Beyond these, we explored knowledge distillation as an alternative compression strategy. We designed *MiDE-Net* (Modified and Distilled ENet), a simplified version of ENet with just 6 bottleneck modules (vs. 25 in ENet), reducing complexity while preserving performance. ENet acted as the teacher, guiding MiDE-Net via soft predictions using a distillation objective based on Kullback-Leibler divergence. Tables 4 and 5 summarize the trade-offs. ENet achieved the highest mIoU on Cityscapes (43.91%) and top detection scores on SMIYC RO-21 and FS Static. MiDE-Net, with only 72K parameters (vs. 354K), retained competitive scores on Road Anomaly (AuPRC: 11.78) and SMIYC RA-21 (AuPRC: 18.50), and ran faster (2.30 fps vs. 1.87 fps). The pruned&quantized MiDENet was most efficient (3.14 fps), but had the lowest performance (mIoU: 14.03%, FS L&F AuPRC: 0.57). These findings show that while pruning and quantization enhance speed and compactness, distillation offers a better balance between accuracy and efficiency.

5. Conclusion

In this study, we investigated real-time anomaly segmentation in road scenes by evaluating multiple lightweight semantic segmentation models and applying several optimization techniques to improve their efficiency and robustness. We began with baseline models—ENet, ERFNet, and BiSeNet—assessed primarily on their anomaly detection capabilities using metrics such as mean IoU, AuPRC, and FPR@95. To improve calibration, we applied temperature scaling, and to better handle out-of-distribution data, we introduced a void classifier trained on void-labeled regions in Cityscapes. Among the models, ERFNet was selected for extensive optimization due to its balance between speed and accuracy. We applied structured L_2-norm pruning and post-training static quantization, both separately and in combination. While pruning effectively reduced the number of active parameters and improved inference speed with limited accuracy degradation, quantization introduced a more noticeable performance drop and increased inference time in some cases, particularly when applied independently. Nevertheless, the combined pruning and quantization configuration offered the best trade-off in terms of memory and computational efficiency. For ENet, we explored knowledge distillation by developing a simplified variant called MiDE-Net. Using the original ENet as the teacher, MiDE-Net was trained to mimic its output distribution through Kullback–Leibler divergence, reducing model complexity while maintaining competitive performance. Despite a natural drop in accuracy due to architectural simplifications, the distilled model showed significant gains in inference speed and parameter reduction, validating the effectiveness of distillation as a compression strategy. Overall, our results suggest that careful application of pruning, quantization, and distillation can yield highly efficient models suitable for real-time anomaly segmentation, with trade-offs that can be tuned according to deployment constraints.

Method	Cityscapes	SMIYC RA-21		SMIYC RO-21		FS L&F		FS Static		Road Anomaly	
	mIoU ↑	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓
MSP	72.20	29.10	62.56	2.71	65.27	1.75	50.62	7.47	41.83	12.67	82.75
MaxLogit	72.20	**38.31**	**59.35**	**4.63**	**48.48**	**3.30**	**45.52**	**9.50**	**40.29**	**15.58**	**73.26**
MaxEntropy	72.20	30.96	62.67	3.04	65.96	2.58	50.19	8.84	62.67	12.42	82.58

Table 1. Performance results of ERFNet across various benchmark datasets for different metrics used as baselines. The table reports mIoU (higher is better), AuPRC (higher is better), and FPR95 (lower is better) metrics, evaluated for different methods: MSP, MaxLogit, and MaxEntropy. Results are evaluated on Cityscapes, SMIYC RA-21, SMIYC RO-21, FS L&F, FS Static, and Road Anomaly datasets.

Method	Cityscapes	SMIYC RA-21		SMIYC RO-21		FS L&F		FS Static		Road Anomaly	
	mIoU ↑	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓
MSP	72.20	29.09	62.56	2.71	65.27	1.75	50.62	7.47	41.83	12.42	82.58
MSP (t = 0.5)	72.20	27.06	62.73	2.42	**63.24**	1.28	66.75	6.60	43.48	12.19	**82.03**
MSP (t = 0.75)	72.20	28.15	**62.50**	2.57	64.15	1.49	51.76	6.99	42.49	12.32	82.32
MSP (t = 1.1)	72.20	29.40	62.66	2.76	65.90	1.86	50.20	7.69	41.62	12.46	82.74
MSP (t = 2.15)	72.20	**30.64**	65.57	**3.01**	73.69	**2.77**	**47.65**	**9.76**	**41.42**	**12.65**	84.88

Table 2. Performance results of ERFNet across various benchmark datasets for the MSP method with various temperatures.

Network	Cityscapes	SMIYC RA-21		SMIYC RO-21		FS L&F		FS Static		Road Anomaly	
	mIoU ↑	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓
ENet (nE = 340)	41.76	15.12	91.62	**4.14**	91.81	5.75	32.90	10.60	65.14	12.89	80.53
ENet (nE = 560)	43.91	13.29	90.85	1.39	76.90	8.07	72.21	11.26	67.53	10.18	89.43
ERFNet	64.31	19.31	78.29	1.41	99.79	10.16	**18.18**	**20.77**	**23.87**	9.71	91.20
BiSeNet	**66.18**	**28.11**	**67.69**	1.82	**39.71**	**12.13**	43.22	18.84	25.31	**13.12**	**74.84**

Table 3. Performance results across datasets of different networks trained as Void Classifiers.

Network	Cityscapes	SMIYC RA-21		SMIYC RO-21		FS L&F		FS Static		Road Anomaly	
	mIoU ↑	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓
ERFNet	**64.31**	**19.31**	**78.29**	1.41	99.79	**10.16**	**18.18**	**20.77**	**23.87**	9.71	91.20
ERFNet (pruned)	58.61	15.61	86.81	0.60	95.41	2.84	34.92	9.37	52.57	7.73	90.77
ERFNet (quantized)	44.80	18.53	82.67	**2.08**	99.82	5.36	31.06	16.78	26.69	**12.00**	91.63
ERFNet (pruned&quantized)	41.06	17.78	82.12	0.62	**95.31**	3.35	57.48	3.12	69.86	8.92	**89.39**
ENet	**43.91**	13.29	90.85	**1.39**	**76.90**	**8.07**	72.21	**11.26**	**67.53**	10.18	89.43
MiDE-Net	25.17	**18.50**	**79.20**	0.69	98.02	1.54	**38.86**	5.55	93.66	**11.78**	**85.86**
MiDENet (pruned&quantized)	14.03	10.64	99.61	0.41	99.75	0.57	62.98	2.62	82.25	7.99	94.39

Table 4. Performance results for ERFNet- and ENet-based Void Classifier models across benchmark datasets.

Network	Inference Time (fps)	FLOPS/IOPS (Billions)	Trainable Parameters (Millions)	Size (MB)
ERFNet	0.22	26.74	2.06	8.02
ERFNet (pruned)	0.67	20.77	1.60	8.02
ERFNet (quantized)	1.62	26.74	-	2.24
ERFNet (pruned&quantized)	**1.76**	20.77	-	2.24
ENet	1.87	4.27	0.354	1.6
MiDENet	2.30	2.45	0.072	0.28
MiDENet (pruned&quantized)	**3.14**	1.74	-	0.12

Table 5. Inference performance comparison for ERFNet- and ENet-based Void Classifier models. Inference time is calculated on the Cityscapes validation set; FLOPS/IOPS are measured neglecting multiplications with zeros.

References

[1] Hermann Blum, Paul-Edouard Sarlin, Juan I. Nieto, Roland Siegwart, and Cesar Cadena. The fishyscapes benchmark: Measuring blind spots in semantic segmentation. *CoRR*, abs/1904.03215, 2019. 4

[2] Robin Chan, Krzysztof Lis, Svenja Uhlemeyer, Hermann Blum, Sina Honari, Roland Siegwart, Mathieu Salzmann, Pascal Fua, and Matthias Rottmann. Segmentmeifyoucan: A benchmark for anomaly segmentation. *CoRR*, abs/2104.14812, 2021. 4, 5

[3] Robin Chan, Matthias Rottmann, and Hanno Gottschalk. Entropy maximization and meta classification for out-of-distribution detection in semantic segmentation. *CoRR*, abs/2012.06575, 2020. 3

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. *CoRR*, abs/1604.01685, 2016. 1, 4

[5] Terrance DeVries and Graham W. Taylor. Learning confidence for out-of-distribution detection in neural networks. 2018. 1

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017. 3

[7] Dan Hendrycks, Steven Basart, Mantas Mazeika, Andy Zou, Joe Kwon, Mohammadreza Mostajabi, Jacob Steinhardt, and Dawn Song. Scaling out-of-distribution detection for real-world settings. 2022. 3

[8] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016. 2

[9] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015. 3

[10] R. Krishnamoorthi. Quantizing deep convolutional networks for efficient inference: A whitepaper. *CoRR*, abs/1806.08342, 2018. 3, 6

[11] Krzysztof Lis, Krishna K. Nakka, Pascal Fua, and Mathieu Salzmann. Detecting the unexpected via image resynthesis. *CoRR*, abs/1904.07595, 2019. 4

[12] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz. Pruning convolutional neural networks for resource-efficient inference. *CoRR*, abs/1611.06440, 2017. 3, 6

[13] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *CoRR*, abs/1606.02147, 2016. 1, 2

[14] Peter Pinggera, Sebastian Ramos, Stefan Gehrig, Uwe Franke, Carsten Rother, and Rudolf Mester. Lost and found: Detecting small road hazards for self-driving vehicles. *CoRR*, abs/1609.04653, 2016. 4

[15] Eduardo Romera, José M. Álvarez, Luis M. Bergasa, and Roberto Arroyo. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263–272, 2018. 1, 2

[16] Jonathon Shlens. Notes on kullback-leibler divergence and likelihood, 2014. 3

[17] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. *CoRR*, abs/1808.00897, 2018. 1, 2

[18] J. Zhang. Fx graph mode post training static quantization, 2021. PyTorch Tutorials. 3