Real-Time Anomaly Segmentation for Road Scenes

Davide Sferrazza
Politecnico di Torino
s326619
s326619@studenti.polito.it

Davide Vitabile
Politecnico di Torino
s330509
s330509@studenti.polito.it

Abstract

Modern Deep Neural Networks, when deployed in open-world settings, perform poorly on Unknown/Anomaly/Out-of-Distribution (OoD) objects that were not present during the training. Detecting OoD objects becomes critical for autonomous driving applications and branches of computer vision problems such as continual learning and open-world problems. In this paper we investigate how to use existing tiny segmentation models to detect anomalies in road scenes. Furthermore, we explore how losses that are specifically made for anomaly detection can boost results. The code used in this paper is available at https://github.com/FarInHeight/Real-Time-Anomaly-Segmentation-for-Road-Scenes/tree/main.

1. Introduction

The detection of anomalies inside an image is a challenging task. Typical methods are based on trained Deep Neural Networks that perform *Semantic Segmentation* and try to discriminate between In-Distribution (ID) samples and Out-of-Distribution (OoD) samples, *i.e.* samples from a different distribution that the network was not exposed to during training and therefore should not be predicted with high confidence at test time.

Since the main applications of anomaly detection, *e.g.* autonomous driving and robot interaction, require fast real-time inference when deployed in real world settings, the trade-off between speed, accuracy and the management of limited resources is a critical aspect.

Our paper focuses on *per-pixel anomaly segmentation* methods, by comparing several techniques which process the outputs of segmentation models [12, 14, 16] trained on Cityscapes [5] and try to inference what pixels are likely to belong to anomalies inside road scenes.

Our first approach was to perform various *anomaly inferences* that provide *pixel-wise anomaly scores* from the output of the models. We then tried to improve the results by employing *temperature scaling*, a confidence calibration method.

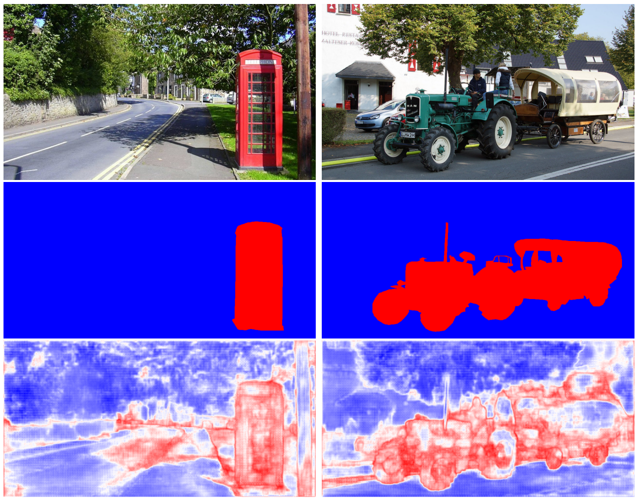


Figure 1. Anomaly scores calculated by an ERFNet model using the MaxLogit inference method. The top row shows the original images, the middle row shows the ground truth anomalies and the bottom row shows the estimated anomaly scores (the hotter the pixel, the higher the associated anomaly score).

Having this starting point, we then explored how performances vary by taking into account Out-of-Distribution knowledge from the networks' outputs, by using a Void Classifier.

In the final part of our experiments, we analyzed the effects of training the models along with losses that are specifically made for the task of anomaly detection.

2. Related Works

Anomaly Segmentation has gained major importance in recent years, due to the need of having networks which give reliable results for the safe deployment in the real world scenarios. The literature has proposed several datasets which serves as baselines to evaluate the goodness of anomaly inference methods, along with techniques and metrics used for measuring the accuracy of the results.

2.1. Datasets and Benchmarks

2.1.1 Cityscapes

Cityscapes [5] is a benchmark suite and large-scale dataset to train and test approaches for pixel-level and instance-level semantic labeling. Cityscapes is composed of a large set of stereo video sequences recorded in streets from 50 different cities. 5000 of these images have high quality pixel-level annotations; 20 000 additional images have coarse annotations to enable methods that leverage large volumes of weakly-labeled data

2.1.2 Fishyscapes

Fishyscapes [2] is the first public benchmark for anomaly detection in a real-world task of semantic segmentation for urban driving. It is comprised of three different datasets, with divergent generation conditions, image content and goals.

FS Static is a dataset based on the validation set of Cityscapes [5]. It is split into a public validation set of 30 images and a hidden test set of 1000 images. It contains approximately 4.5×10^7 OoD and 1.8×10^9 ID pixels. It is created with a limited visual diversity, which is important to make sure that it contains none of the overlayed objects.

FS Web is a dynamically changing dataset built in similar fashion to FS static, but the overlaid objects are crawled from the internet using a changing list of keywords. Its purpose is to measure any possible overfitting of the anomaly detection methods.

FS Lost and Found is a dataset based on the original Lost and Found [13] dataset, which contains pixel-wise annotations that distinguish between *objects* (the anomalies), *background* (classes contained in Cityscapes) and *void* (anything not contained in Cityscapes classes that still appears in the training images). It consists of 100 validation set images and a test set of 275 images, based on disjoint sets of locations. The dataset allows to use real images for testing, therefore preventing any form of overfitting on synthetic image processing.

2.1.3 Road Anomaly

Road Anomaly [10] is a collection of online images depicting anomalous objects located on or near the road. It consists of 60 images of size 1280×720.

2.1.4 SegmentMeIfYouCan

SegmentMeIfYouCan [3] is a popular benchmark which addresses two tasks: anomalous object segmentation, which considers any object category that has not been seen before; and road obstacle segmentation, which focuses on any object located on the road, whether known or unknown. It contains two different datasets, that differ from each other based on where the anomalies appear in the images.

RoadAnomaly21 is a dataset comparable to FS Lost and Found [2]. The anomalies can appear anywhere in the image, with each image showing a unique real scene where at least one anomalous object appears. Each anomalous sample widely differs in size.

RoadObstacle21 is a dataset similar to Lost and Found [13]. All anomalies (or obstacles) appear on the road, and each image shows a scene in a particular situation, including night, dirty roads and snowy conditions.

2.2. Architectures

The task of *Real-Time Anomaly Segmentation* requires lightweight networks capable of producing reliable results while limiting the resources employed and keeping the speed of inference high. Various tiny, high-performance architectures have been proposed in the literature, providing special computational tricks or enlightening insights.

2.2.1 ENet

ENet (Efficient Neural Network) [12] is a network designed for tasks requiring low-latency operations, as the network is particularly fast in making predictions.

ENet is an asymmetric encoder-decoder network, whose main component is a bottleneck module as illustrated in Fig. 2.

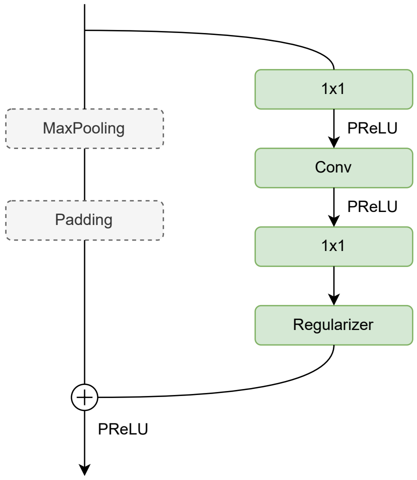


Figure 2. ENet bottleneck module. `Conv` is either a regular, dilated, or transposed convolution with 3×3 filters, or a 5×5 convolution decomposed into two asymmetric ones. In the decoder, max pooling is replaced with max unpooling, and padding is replaced with spatial convolution without bias.

2.2.2 ERFNet

ERFNet (Efficient Residual Factorized ConvNet) [14] is a real-time and accurate semantic segmentation convolutional neural network.

Like ENet, ERFNet is an asymmetric encoder-decoder network whose core component is a Non-bottleneck-1D block whose structure is shown in Fig. 3. The skip connections inside the block allows the convolutions to learn residual functions that facilitate the training. The 1D factorized convolutions allow a substantial reduction of the computational cost while preserving a similar accuracy compared to the 2D ones.

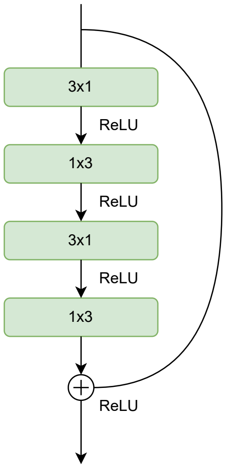


Figure 3. ERFNet Non-bottleneck-1D module.

2.2.3 BiSeNet

BiSeNet (Bilateral Segmentation Network) [16] is a real-time semantic segmentation network with a balanced combination of speed and segmentation performance. This network architecture seeks to overcome the constraints dictated by modern approaches that usually compromise spatial resolution to achieve real-time inference speed, resulting in poor performance.

To make a good trade-off between these two aspects, BiSeNet uses two main branches called **Spatial Path** and **Context Path** (Fig. 4a). The goal of the Spatial Path is to preserve spatial information for generating high-resolution features, while the Context Path aims to rapidly downsample the input image to obtain a sufficiently large receptive field.

The network employs two specially crafted modules:

- **Attention Refinement Module**: the module, indicated in Fig. 4b, computes an attention vector from a feature map and re-weights the original features by the attention vector, capturing the global context information without the need of complex up-sample operations;
- **Feature Fusion Module**: the module combines the features of the Spatial Path with the features of the Context Path, keeping in mind that the two paths encodes information at different levels. The module structure is illustrated in Fig. 4c.

3. Methods

Pixel-level Anomaly Segmentation can be tackled in several ways. To understand how well tiny segmentation models perform anomaly inferences, we adopted a pre-trained ERFNet model on 19 Cityscapes classes and tested its performance on three different standard evaluation methods [3]. The datasets on which we performed the tests are Road-Anomaly21, RoadObstacle21, FS Lost and Found, FS Static and Road Anomaly.

3.1. Baselines

For mathematical notation, we use $f(\boldsymbol{x};\theta)$ to denote the output of a network for an input $\boldsymbol{x}$, also known as the logit, where θ are the network parameters. Let $\mathcal{Z}$ indicate the set of image coordinates and let $\sigma : \mathbb{R}^{|\mathcal{C}|} \to [0,1]^{|\mathcal{C}|}$ denote the softmax function with $\mathcal{C}$ indicating the set of all classes. The problem of pixel-level anomaly segmentation can be formulated as a binary classification task: determining whether a pixel is OoD or not can be performed using a simple threshold $\delta \in \mathbb{R}$, where a pixel at location $z \in \mathcal{Z}$ is considered an anomaly if its anomaly score $s_z(\boldsymbol{x})$ is such that $s_z(\boldsymbol{x}) \geq \delta$. The baseline methods we used are the following:

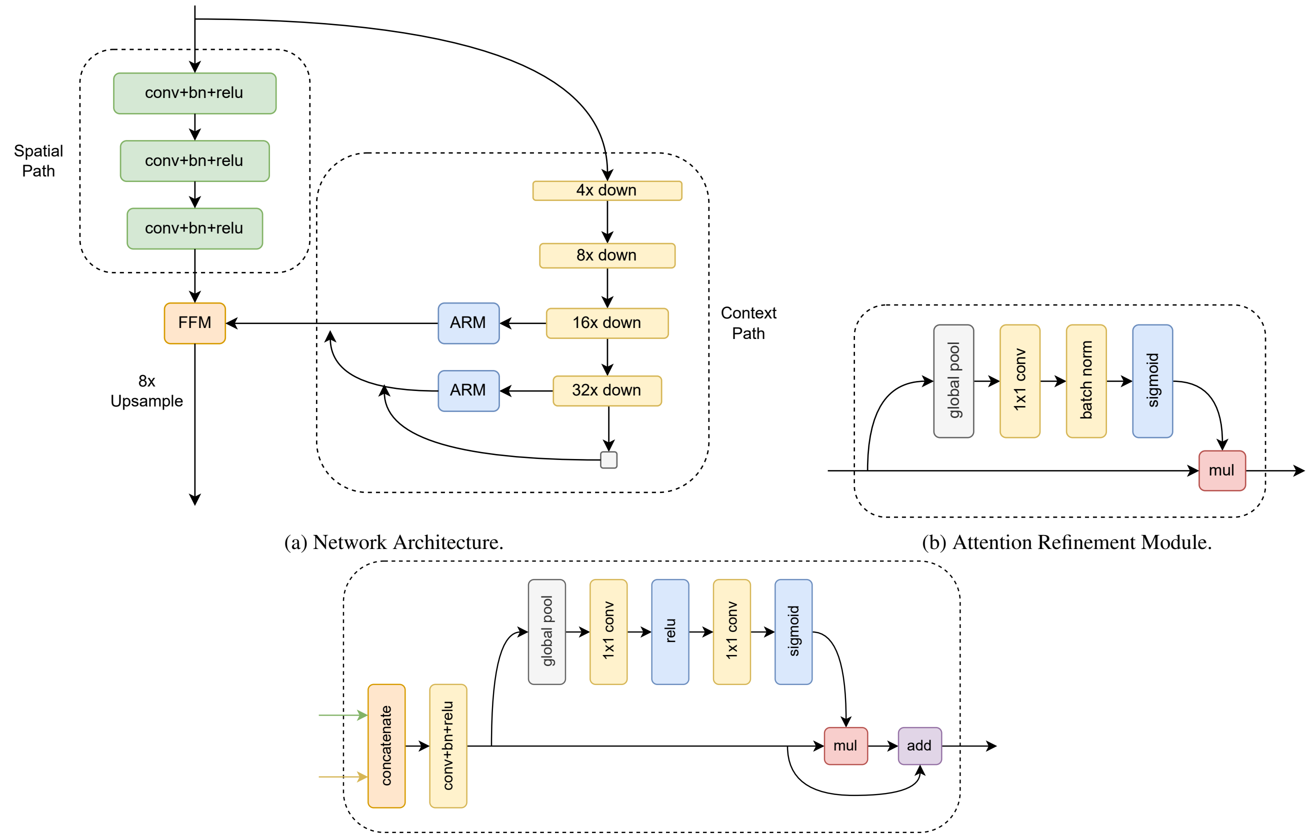
MSP (Maximum Softmax Probability) [8]: This inference method is based on the idea that correctly classified ID samples tend to have larger maximum softmax probabilities than misclassified OoD samples. The anomaly score of a particular pixel at location $z \in \mathcal{Z}$ is computed as:

$$s_z(\boldsymbol{x}) = 1 - \max_{c \in \mathcal{C}} \sigma(f_z^c(\boldsymbol{x};\theta)) \tag{1}$$

MaxLogit [7]: The preceding inference method is problematic when dealing with ID datasets containing a large number of classes because the probability mass can be distributed among visually similar classes. To overcome this issue, MaxLogit computes the anomaly score of a particular pixel at location $z \in \mathcal{Z}$ using the unnormalized logits as:

$$s_z(\boldsymbol{x}) = -\max_{c \in \mathcal{C}} f_z^c(\boldsymbol{x};\theta) \tag{2}$$

Maximized Entropy (Max Entropy) [4]: To take into account the uncertainty of the probability distribution generated by the softmax function calculated



(a) Network Architecture.

(b) Attention Refinement Module.

(c) Feature Fusion Module.

Figure 4. BiSeNet.

from the logits, Max Entropy computes the anomaly score of a particular pixel at location $z \in \mathcal{Z}$ using the calculation of the distribution entropy:

$$s_z(\boldsymbol{x}) = -\sum_{c \in \mathcal{C}} \sigma(f_z^c(\boldsymbol{x};\theta)) \log(\sigma(f_z^c(\boldsymbol{x};\theta))) \quad (3)$$

The goodness of inference methods for anomaly segmentation is typically evaluated through two holistic metrics: the *area under the precision-recall curve (AuPRC)* and the *false positive rate at* 95% *true positive rate (FPR95)*. The AuPRC measure is calculated by considering the precision and recall as functions of some threshold $\delta \in \mathbb{R}$ applied to the anomaly scores. AuPRC provides a suitable metric for class imbalances, which is relevant in anomaly detection, when the number of anomalies may be relatively small. The FPR95 measure, on the other hand, indicates how many false positive predictions should be made to achieve the desired true positive rate. Typically, along with the anomaly inference results, we add the *mean Intersection-over-Union (mIoU)* values, which is a standard metric used to assess the performance of semantic segmentation networks. The *Intersection-over-Union (IoU)* is computed as:

$$IoU = \frac{TP}{TP + FP + FN} \quad (4)$$

where TP, FP and FN are the number of true positives, false positives and false negatives at pixel level, respectively. The global mIoU value is obtained by averaging the IoU class values computed over a batch of images.

The results we obtained are given in Table 1. As can be seen from the results, the MaxLogit method consistently outperforms the other methods on all datasets. The MSP method performs slightly worse on average than the Max Entropy method. This result is due to the fact that the Max Entropy method leverages the uncertainty in the probability distribution of the softmax output to calculate the anomaly scores and improve the results. Its results could be further improved by fine-tuning the ERFNet model with a multi-criteria loss function that includes the averaged negative log-likelihood of out-of-distribution data. Overall, MaxLogit has the ability to better discriminate between visually similar classes and is therefore able to more accurately predict whether the pixel belongs to an anomaly or not.

Method	mIoU	SMIYC RA-21		SMIYC RO-21		FS L&F		FS Static		Road Anomaly	
		AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓
MSP	72.20	29.09	62.587	2.714	64.984	1.748	50.788	7.427	41.837	12.432	82.516
MaxLogit	72.20	**38.287**	**59.436**	**4.629**	**48.454**	**3.299**	**45.532**	**9.436**	**40.318**	**15.555**	**73.322**
Max Entropy	72.20	30.992	62.679	3.054	65.569	2.58	50.424	8.768	41.558	12.679	82.639

Table 1. ERFNet anomaly inferences on RoadAnomaly21, RoadObstacle21, FS Lost and Found, FS Static and Road Anomaly. All values are indicated as percentages. ↑ means larger values are better, and ↓ means smaller values are better. **Bold numbers** indicate the best results.

4. Experiments

4.1. Temperature Scaling

Temperature Scaling [6] is a confidence calibration method, *i.e.* a method that tries to solve the problem of predicting probability estimates associated with the predicted class label which are representative of the true correctness likelihood. We investigate how the MSP inference method is affected by temperature scaling, choosing as anomaly score of a particular pixel at location $z \in \mathcal{Z}$ the following value:

$$s_z(\boldsymbol{x}) = 1 - \max_{c \in \mathcal{C}} \sigma(f^c_z(\boldsymbol{x};\theta)/T) \tag{5}$$

where T is the temperature, a floating-point number. In Tab. 2, we show results for some temperature values in range $[0, 1.1]$ and in the last row the best trade-off temperature value that we found. By performing various anomaly inferences using this method and varying the temperature value, we found that the best performance on all datasets is obtained with values in range $[1, 2]$, so we defined a search grid and explored this interval. We chose $T = 1.6$ as the best trade-off temperature value because, on average, for temperature values $T > 1.6$, the results for the RoadAnomaly21, RoadObstacle21 and Road Anomaly datasets continue to worsen, while the results for the FS Lost and Found and FS Static datasets continue to improve.

4.2. Void Classifier

After analyzing the anomaly segmentation performance on some baseline methods, we dove into understanding how results are affected if we take into account also the void class. The metric numbers shown in Tabs. 1 and 2 are obtained by discarding the void class output, *i.e.* the 20th class defined in the Cityscapes dataset, also known as background class. Considering the void class output as an anomaly indicator, we can compute an anomaly score of a particular pixel at location $z \in \mathcal{Z}$ as the following:

$$s_z(\boldsymbol{x}) = \sigma(f^{\text{void}}_z(\boldsymbol{x};\theta)) \tag{6}$$

The obtained classifier is called *Void Classifier* [1].

We performed the tests on three different architectures: ENet, ERFNet and BiSeNet (the training details are explained in Section 4.4). The results are shown in Tab. 3. From the values obtained, we can see that ERFNet outperforms the other architectures on the FS Lost&Found and FS Static datasets, where the model gives much better FPR95 values than the other architectures (improvements of about 30% and 50%). For the other datasets, BiSeNet performs best on average. For the RoadAnomaly21 dataset, BiSeNet achieves an impressive direct improvement of about 30% over the other models.

4.3. Loss Functions

When training Deep Neural Networks for the task of semantic segmentation, the typical choice of loss function to use is the well-known **Cross-Entropy (CE) Loss**. The network produces logits, which are the input of a softmax function σ, and the loss of a particular pixel at location $z \in \mathcal{Z}$ is computed as:

$$\mathcal{L}_{\text{CE}} = -\log(\sigma(f^t_z(\boldsymbol{x};\theta))) \tag{7}$$

where t is the ground truth class of the pixel.

To focus the training on hard examples, Lin *et al.* [9] devised a new training loss called **Focal Loss (FL)**. In its general form, the Focal Loss of a particular pixel at location $z \in \mathcal{Z}$ is calculated as:

$$\mathcal{L}_{\text{FL}} = -\alpha_t \left(1 - \sigma(f^t_z(\boldsymbol{x};\theta))\right)^{\gamma} \log(\sigma(f^t_z(\boldsymbol{x};\theta))) \tag{8}$$

where α_t is a class-dependent weight used for addressing class imbalance and γ is a tunable *focusing* parameter. This loss scales down the contribution of simple examples and quickly shifts the focus of model training towards hard examples.

In our experiments, we modified the logits of the ERFNet model according to the **Enhanced Isotropy Maximization Loss** and **Logit Normalization** method. Macêdo *et al.* [11] replaced the last linear layer of a classifier network to compute logit for class c as follows:

$$f^c_L(\boldsymbol{x};\theta) = -\left|d_s\right| \left\| \widehat{f_{L-1}(\boldsymbol{x};\theta)} - \widehat{p^c(\phi)} \right\| \tag{9}$$

where $\widehat{p^c(\phi)}$ is a normalized *learnable prototype* associated with class c, $\widehat{f_{L-1}(\boldsymbol{x};\theta)}$ is the normalized output of the second-to-last network layer and d_s is a *scalar learnable*

Method	mIoU	SMIYC RA-21		SMIYC RO-21		FS L&F		FS Static		Road Anomaly	
		AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓
MSP $(T = 1)$	72.20	29.09	62.587	2.714	64.984	1.748	50.788	7.427	41.837	12.432	82.516
MSP $(T = 0.5)$	72.20	27.05	62.805	2.422	**63.209**	1.28	66.78	6.562	43.489	12.194	**82.032**
MSP $(T = 0.75)$	72.20	28.144	62.557	2.567	64.088	1.493	51.86	6.953	42.496	12.325	82.305
MSP $(T = 1.1)$	72.20	29.398	62.681	2.769	65.51	1.859	50.431	7.641	41.617	12.471	82.629
MSP $(T = 1.6)$	72.20	**30.373**	**63.623**	**2.968**	68.609	**2.378**	**49.247**	**8.733**	**41.046**	**12.616**	83.417

Table 2. ERFNet anomaly inferences using temperature scaling on RoadAnomaly21, RoadObstacle21, FS Lost and Found, FS Static and Road Anomaly. All values are indicated as percentages. ↑ means larger values are better, and ↓ means smaller values are better. **Bold numbers** indicate the best results.

Method	mIoU	SMIYC RA-21		SMIYC RO-21		FS L&F		FS Static		Road Anomaly	
		AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓
ENet	58.20	12.602	90.674	0.777	**70.262**	1.995	55.395	6.47	78.233	8.089	94.431
ERF-Net	72.50	17.924	**79.127**	1.028	94.319	**8.263**	**31.719**	**10.275**	**39.894**	10.422	**80.666**
BiSeNet	68.78	**45.819**	89.569	**11.414**	85.895	8.246	69.243	7.523	93.824	**15.135**	88.114

Table 3. Anomaly inferences using a Void Classifier on ENet, ERFNet and BiSeNet on RoadAnomaly21, RoadObstacle21, FS Lost and Found, FS Static and Road Anomaly datasets. ENet and ERFNet were fine-tuned for 20 epochs, while BiSeNet was trained from scratch for 300 epochs. All values are indicated as percentages. ↑ means larger values are better, and ↓ means smaller values are better. **Bold numbers** indicate the best results.

parameter called *distance scale* which is used to avoid unreasonable restrictions on distance values. The *Enhanced Isotropy Maximization Loss (IsoMax+)* is then calculated as:

$$\mathcal{L}_{\text{IsoMax+}} = -\log(\sigma(E_s f_L^t(\boldsymbol{x};\theta))) \tag{10}$$

where E_s represents the entropic scale. Since ERFNet is an encoder-decoder semantic segmentation network whose output is a $20 \times H \times W$ tensor, the method is not directly applicable as such. As the network output is given by a feature vector for each pixel of an image, we decided to modify the method by making the following changes:

1. $p^c(\phi)$ is a *scalar learnable prototype* instead of a vector;
2. the logit for class c and pixel at location $z \in \mathcal{Z}$ is computed from the last layer as
$$f_z^c(\boldsymbol{x};\theta) = -|d_s|\left|f_{z,L}^c(\boldsymbol{x};\theta) - p^c(\phi)\right| \tag{11}$$
3. the loss for each pixel is therefore given by:
$$\mathcal{L}_{\text{IsoMax+}} = -\log(\sigma(E_s f_z^t(\boldsymbol{x};\theta))) \tag{12}$$

The method of Wei *et al*. [15], on the other hand, is based on the observation that the Cross-Entropy Loss can further increase the magnitude of the logit vector leading to an overconfidence issue even though the examples are already correctly classified. This method encourages the direction of the logit vector to be consistent with the corresponding one-hot label, without optimizing the magnitude of the vector as there is no need to do so. The *Logit Normalization (LogitNorm)* method is simply implemented by dividing the output logit vector of a classifier network by its norm so that the Cross-Entropy loss is given by:

$$\mathcal{L}_{\text{logit\_norm}} = -\log\left(\sigma\left(\frac{f^t(\boldsymbol{x};\theta)}{T\,\|f(\boldsymbol{x};\theta)\|}\right)\right) \tag{13}$$

where T is the temperature value that modulates the magnitude of the logit vector. To adapt the method for the semantic segmentation task, we normalized the output logit vector of each pixel separately, so that the loss of a pixel at location $z \in \mathcal{Z}$ is calculated as:

$$\mathcal{L}_{\text{logit\_norm}} = -\log\left(\sigma\left(\frac{f_z^t(\boldsymbol{x};\theta)}{T\,\|f_z(\boldsymbol{x};\theta)\|}\right)\right) \tag{14}$$

At test time, the logit vector is not normalized.

We propose an analysis of the performance of an ERFNet architecture (the training details are explained in Section 4.4) when IsoMax+ and LogitNorm are used together

Method	Loss	mIoU	SMIYC RA-21		SMIYC RO-21		FS L&F		FS Static		Road Anomaly	
			AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓	AuPRC ↑	FPR95 ↓
MSP	CE	67.78	18.356	71.601	**16.267**	22.418	1.437	66.566	7.152	45.441	13.213	77.188
	LN + CE	67.82	24.542	73.381	12.012	19.247	0.663	82.083	6.042	67.586	13.737	72.309
	LN + FL	62.23	18.549	84.332	3.982	26.469	**2.45**	**65.172**	**16.599**	**23.813**	12.499	76.645
	IsoMax + CE	63.63	23.073	**63.369**	7.205	30.517	1.255	74.88	9.798	55.366	14.105	67.171
	IsoMax + FL	61.15	**29.186**	77.702	4.419	**15.527**	1.863	81.091	13.088	34.858	**15.612**	**66.703**
MaxLogit	CE	67.78	19.305	**69.209**	**13.174**	**18.507**	3.147	71.692	6.578	54.075	**14.655**	73.781
	LN + CE	67.82	**23.04**	76.728	10.448	22.741	0.785	82.082	6.391	67.673	14.459	**72.895**
	LN + FL	62.23	21.999	93.995	9.608	77.415	0.929	71.512	8.251	59.84	14.265	90.158
	IsoMax + CE	63.63	13.533	95.804	1.159	90.02	0.407	92.547	1.717	94.606	9.256	93.555
	IsoMax + FL	61.15	22.815	87.177	2.926	21.864	**3.951**	**50.865**	**23.126**	**29.211**	13.425	75.345
Max Entropy	CE	67.78	18.085	71.91	**16.085**	22.154	**1.87**	**65.87**	**7.937**	**45.295**	13.331	76.969
	LN + CE	67.82	19.064	83.612	5.89	49.558	0.359	82.13	2.672	79.669	11.511	79.251
	LN + FL	62.23	12.009	99.137	0.728	95.193	0.794	72.904	7.041	66.705	11.113	93.605
	IsoMax + CE	63.63	**24.385**	**70.101**	4.944	32.503	0.545	67.226	4.58	78.172	14.702	**68.443**
	IsoMax + FL	61.15	21.002	96.586	7.219	**16.364**	0.745	73.068	6.515	55.82	**18.732**	76.865

Table 4. ERFNet anomaly inferences on RoadAnomaly21, RoadObstacle21, FS Lost and Found, FS Static and Road Anomaly. The trainings were carried out for 100 epochs (IsoMax is used here to refer to the IsoMax+) by setting $\gamma = 2, \alpha_c = 1 \, \forall c \in C, E_s = 10$ and $T = 0.04$. All values are indicated as percentages. ↑ means larger values are better, and ↓ means smaller values are better. **Bold numbers** indicate the best results for each method. **Blue and bold numbers** indicate the best results among all methods.

with Cross-Entropy and Focal Loss to train the model. The results are shown in Table 4. As can be seen from the results, the IsoMax+ and LogitNorm losses generally perform better than the Cross-Entropy loss when using the MSP inference method on all datasets. Specifically, for the MSP method, we can conclude that the best results are given by the ad hoc anomaly detection losses when used together with the Focal Loss, although the obtained mIoU are lower. On the other hand, MaxLogit and Max Entropy give poor performance compared to the results we obtained in Table 1, except for the FS Lost&Found and FS Static datasets, for which MaxLogit outperforms the other methods when IsoMax+ is used together with Focal Loss. We believe that these losses do not produce results as good as those shown in Tab. 1 for MaxLogit because the losses we used aim to modify the outputs of the softmax function by operating directly on the logit vectors produced by the networks. MaxLogit needs un-normalized logit outputs to better discriminate between visually similar classes, so it goes against what the Logit Normalization method does.

4.4. Training

To obtain the results shown in Table 3, we used a pre-trained ERFNet[1] and ENet[2] on 19 Cityscapes classes. We then performed fine-tuning for 20 epochs by including the *unlabeled* class using the training configurations found on the owners' GitHub repositories, but starting from a learning rate 10 times smaller. We also trained a BiSeNet[3] model from scratch trying to follow the original paper of Changqian *et al.* [16]. We used a SGD (Stochastic Gradient Descent) optimizer with a batch size of 16 images, momentum 0.9 and weight decay 10^{-4}. We applied a polynomial learning rate decay strategy, so that the learning rate is multiplied by $\left(1 - \frac{epoch}{\#epochs}\right)^{power}$ after each epoch with power fixed to 0.9. Data pre-processing was performed by dividing each channel by 255, employing per-channel mean subtraction and division by standard deviation. Each image was augmented with a random horizontal flip, random scale and random crop. The scale was chosen from $\{0.75, 1.0, 1.5, 1.75, 2.0\}$. The training was carried out for 300 epochs on final images of fixed size equal to 1024×512 using an unweighted Cross-Entropy loss. We did not reach the same results of the original paper probably due to a mismanagement of the learning rate decay strategy and because we trained on 20 instead of 19 Cityscapes classes without using a weighted Cross-Entropy loss.

To fill in Table 4, we trained several ERFNet models from scratch using the Logit Normalization method and the Enhanced Isotropy Maximization method jointly with Cross-Entropy loss and Focal Loss. For a fair comparison, we also performed a new training from scratch of the ERFNet architecture using a weighted Cross-Entropy loss. The trainings were carried out for 100 epochs (50 for the encoder only and 50 for the whole model) using the same hyperparameters found in the authors' GitHub repository and by keeping at the end the best models found during the training sessions. For the Focal Loss we used $\gamma = 2$ and $\alpha_c = 1 \, \forall c \in \mathcal{C}$, for the IsoMax+ loss we used $E_s = 10$ and for the LogitNorm loss we used $T = 0.04$.

[1] https://github.com/Eromera/erfnet_pytorch

[2] https://github.com/zh320/realtime-semantic-segmentation-pytorch

[3] https://github.com/CoinCheung/BiSeNet

5. Conclusions

In this work, we investigated the performance of tiny segmentation models on the task of anomaly segmentation. First, we analyzed how well-known models such as ERFNet behave when baseline inference methods are used on top of them. We also tried to improve the results obtained using a confidence calibration method called *temperature scaling*. All of these methods do not require *out-of-distribution (OoD)* data to train or fine-tune models, making them suitable for data-shortage situations. We then delved into understanding how performance changes when using OoD data to tune models for training *Void Classifiers*. For this task, we fine-tuned pre-trained models and trained a BiSeNet model from scratch. Finally, we experimented with losses specifically designed for anomaly detection, such as the Enhanced Isotropy Maximization loss and the Logit Normalization loss. The results in this paper can certainly be improved by fine-tuning the ERFNet pre-trained model for the Max Entropy method and by training ENet and ERFNet architectures from scratch with the goal of developing better Void Classifier models. The Void Classifier based on BiSeNet can be further improved by refining its training, with the goal of obtaining a result close to the official paper. Finally, future improvements could include training ERFNet and also the other architectures with specifically designed anomaly detection losses for more epochs and examining their results in more detail when trained with different hypermeter values.

References

[1] Hermann Blum, Paul-Edouard Sarlin, Juan Nieto, Roland Siegwart, and Cesar Cadena. Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, Oct 2019. 5

[2] Hermann Blum, Paul-Edouard Sarlin, Juan Nieto, Roland Siegwart, and Cesar Cadena. The fishyscapes benchmark: Measuring blind spots in semantic segmentation. *International Journal of Computer Vision*, 129:3119–3135, 2021. 2

[3] Robin Chan, Krzysztof Lis, Svenja Uhlemeyer, Hermann Blum, Sina Honari, Roland Siegwart, Pascal Fua, Mathieu Salzmann, and Matthias Rottmann. Segmentmeifyoucan: A benchmark for anomaly segmentation. *arXiv preprint arXiv:2104.14812*, 2021. 2, 3

[4] Robin Chan, Matthias Rottmann, and Hanno Gottschalk. Entropy maximization and meta classification for out-of-distribution detection in semantic segmentation. In *Proceedings of the ieee/cvf international conference on computer vision*, pages 5128–5137, 2021. 3

[5] 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. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016. 1, 2

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017. 5

[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. *arXiv preprint arXiv:1911.11132*, 2019. 3

[8] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *arXiv preprint arXiv:1610.02136*, 2016. 3

[9] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 5

[10] Krzysztof Lis, Krishna Nakka, Pascal Fua, and Mathieu Salzmann. Detecting the unexpected via image resynthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2152–2161, 2019. 2

[11] David Macêdo and Teresa Ludermir. Enhanced isotropy maximization loss: Seamless and high-performance out-of-distribution detection simply replacing the softmax loss. *arXiv preprint arXiv:2105.14399*, 2021. 5

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

[13] Peter Pinggera, Sebastian Ramos, Stefan Gehrig, Uwe Franke, Carsten Rother, and Rudolf Mester. Lost and found: detecting small road hazards for self-driving vehicles. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1099–1106. IEEE, 2016. 2

[14] 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, 3

[15] Hongxin Wei, Renchunzi Xie, Hao Cheng, Lei Feng, Bo An, and Yixuan Li. Mitigating neural network overconfidence with logit normalization. In *International Conference on Machine Learning*, pages 23631–23644. PMLR, 2022. 6

[16] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 325–341, 2018. 1, 3, 7