FSNet: A Failure Detection Framework for Semantic Segmentation

Quazi Marufur Rahman, Niko S"underhauf, Peter Corke and Feras Dayoub

Abstract—Semantic segmentation is an important task that helps autonomous vehicles understand their surroundings and navigate safely. However, during deployment, even the most mature segmentation models are vulnerable to various external factors that can degrade the segmentation performance with potentially catastrophic consequences for the vehicle and its surroundings. To address this issue, we propose a failure detection framework to identify pixel-level misclassification. We do so by exploiting internal features of the segmentation model and training it simultaneously with a failure detection network. During deployment, the failure detector flags areas in the image where the segmentation model has failed to segment correctly. We evaluate the proposed approach against state-of-the-art methods and achieve 12.30%, 9.46%, and 9.65% performance improvement in the AUPR-Error metric for Cityscapes, BDD100k, and Mapillary semantic segmentation datasets.

I. INTRODUCTION

Semantic segmentation using deep learning has become crucial for many safety-critical systems such as vision-based self-driving cars [1], [2] and robot-assisted surgery [3], [4]. For instance, semantic segmentation is a significant component for any self-driving car for safety, reliability, and scene understanding [5], [6]. Besides, it plays a substantial role in navigation [7], [8] and obstacle avoidance [9], [10] by segmenting critical objects such as pedestrians and other vehicles in real-time from the visual sensory. Due to its importance, there is ongoing research [11]–[13] to improve the overall performance of semantic segmentation to meet the safety-critical demand in robotic vision.

State-of-the-art research in semantic segmentation commonly assumes that the images encountered during training and later during deployment follow a similar distribution. However, this cannot be guaranteed for applications on autonomous vehicles that operate in the open, unconstrained world. The segmentation model will inevitably encounter situations (objects, object configurations, textures), environmental conditions (weather), or imaging conditions (motion blur, illumination, and exposure effects) that were never seen during training. As a result, a severe drop in segmentation performance could occur without prior warning, posing an extreme risk for the vehicle and its surroundings.

The ideal solution to achieve consistent semantic segmentation performance in all conditions is a highly effective, robust, and domain agnostic model trained using all possible scenarios that it will encounter during the deployment phase. However, these requirements are infeasible for a practical scenario. Another approach is to identify and remove inputs that can decrease segmentation accuracy. Out-of-distribution [14] and open-set [15] detection are examples of this approach. However, these approaches still cannot scale to the complex semantic scenes and structure in which an autonomous vehicle operates [16].

Similarly, uncertainty and confidence estimation can be used to detect incorrect semantic segmentation. However, recent works from [16] and [17] show that these approaches alone are not effective enough to detect perception failure in semantic segmentation. Instead, [16], [18], [19] have argued in favor of using a specifically trained model to identify the incorrect perception of a target model without depending on approaches such as out-of-distribution, open-set, novelty detection, uncertainty, and confidence estimation.

In the context of semantic segmentation, several approaches such as failure prediction [16], [20], introspective perception [17], [18], and quality prediction [21]–[23] train a separate model to identify the semantic segmentation failure. Most of these works are not applicable in autonomous vehicle scenarios, considering the complexity and significant variance of visual sensor inputs encountered by the semantic segmentation network during the deployment. Moreover, these works separately train the segmentation network deployed on an autonomous vehicle may fail to predict the correct label in an input image for various reasons. Our proposed failure detection network is trained to identify the mismatch between prediction and ground truth. Here (a) and (b) respectively show input and output of the semantic segmentation network (c) A binary map showing the mismatch between ground truth and segmentation prediction. (d) Output from our failure detection network. The red colour highlights where the failure detection network identifies inaccurate prediction by the semantic segmentation network.

The authors are with Queensland University of Technology (QUT), Brisbane, Australia. We acknowledge the ongoing support from QUT’s Centre for Robotics. Contact: quazimarufur.rahman@hdr.qut.edu.au
and the failure detection network from each other. Hence, the failure detection network can not access the potentially rich convolutional features of the segmentation network. Besides, a failure-dataset [16], [17] is used to train the failure detection network. The failure-dataset is generated from a relatively smaller holdout set of the initial semantic segmentation training dataset. For example, [17] uses 20% of the segmentation dataset as failure-dataset to train the failure detection network. Therefore, the failure detector cannot exploit the available segmentation dataset and may lack generalizability.

This paper proposes a novel framework, FSNet, consisting of semantic segmentation and a corresponding failure detection network. In contrast to existing works, both of these networks are connected and trained simultaneously. This architecture allows the failure detector to exploit the internal convolutional features of the semantic segmentation network, leading to better failure detection performance. Besides, FSNet uses a joint learning technique to train both networks using the available semantic segmentation dataset. Hence, we can train the failure detection network using the entire segmentation dataset rather than relying on a failure-dataset. We evaluate FSNet against the current SOTA methods using multiple datasets representing in and out-distribution scenarios. Although the segmentation and failure detection networks are connected and trained jointly, our framework does not impede the segmentation accuracy. Our experimental results show that the accuracy of our jointly trained segmentation network is similar to a separately trained network. At the same time, the jointly trained failure detection network outperforms all existing approaches. Figure 1 shows an example of semantic segmentation, the mismatch between the predicted segmentation and ground-truth, and how FSNet can identify that incorrect segmentation.

II. RELATED WORKS

Failure detection or introspection is an essential requirement in robotics to ensure safety and reliability [24]. Morris [25] first proposed a robotic introspection framework to monitor operational state of robot for decision making purpose. Later [26], [27] extended this idea for semantic mapping and obstacle avoidance in robotics. Zhang et al. [18] introduced alert – a framework that predicts the failure of another model. A similar approach has been used by [19] for failure prediction in MAV, hardness predictor [28] for image classifier, and performance monitoring [23] for robot perception system. In this work, we focus on detecting the failure of a semantic segmentation model in the autonomous vehicle context.

The study of failure detection or identifying the erroneous prediction of a model is closely related to uncertainty and confidence estimation. Hendrycks et al. [29] used Maximum Softmax Probability (MSP) derived from the softmax layer for detecting a failure in classification tasks. This work is considered as the standard baseline in related literature. However, MSP suffers drawbacks such as failure to distinguish between in and out-distribution samples and improper calibration. To reduce the risk of making incorrect classification, Geifman et al. [30] introduced selective classifier. This approach controls and guarantees the risk level of a classifier by using thresholds on pre-defined confidence functions, e.g., MSP. Heinrich et al. [31] introduced trust score that compares the prediction between a classifier and a modified nearest neighbor classifier to measure classifier reliability. Most recently, Corbiere et al. [20] has presented true class probability to improve the unreliable ranking of confidence score. Besides, MC Dropout based techniques have become popular for failure detection in classification. However, Xia et al. [16] has argued that these approaches are not applicable in semantic segmentation because of the lack of information on semantic structure and contexts.

Failure detection in the context of semantic image segmentation is being studied extensively in recent years. Kohlberger et al. [32] used a novel space of segmentation features to predict overlap error and the Dice coefficient of an organ segmentation model. Later, Valindria et al. [33] have introduced reverse classification accuracy to predict segmentation quality of medical image segmentation. Huang et al. [34] showed that segmentation quality could be predicted using QualityNet. [21], [35] showed the application of Bayesian CNN for predicting semantic segmentation failure. [36], [37] used unsupervised learning to quantify the quality of semantic segmentation tasks. However, because providing image-level segmentation quality rather than pixel-level failure detection, these works do not apply to identifying the areas where semantic segmentation is incorrect.

Xia et al. [16] have introduced SynthCP to predict pixel-level failure in semantic segmentation. They also demonstrated the usage of [17], [20], [29], [38] for a similar task. Here, [16] and [17] do not access the internal convolutional features of the segmentation network during the training or inference phase. Besides, these works explicitly use a failure-dataset to train the failure detector and consequently can not exploit the entire segmentation dataset. [20], [29], [38] use indirect approaches of using segmentation confidence and entropy for failure detection, which is suboptimal [16], [17].

Our proposed framework FSNet focuses on addressing the shortcomings of the current research. Firstly, this framework enables the failure detector to access the convolutional features of the segmentation network. Secondly, FSNet jointly trains its networks using the entire segmentation dataset without requiring the explicit failure-dataset. Both of these techniques contribute to the better performance of FSNet than all other existing approaches.

III. APPROACH OVERVIEW

This section introduces our failure detection framework FSNet for semantic segmentation. FSNet consists of two connected components – one semantic segmentation network and one failure detection network. We will describe both of these networks and how they work jointly to detect the failure of semantic segmentation.
Fig. 2: An outline of FSNet framework, consisting of the semantic segmentation and failure detection network. It shows the inter-connection between the segmentation and failure detection network.

### A. Module Architecture

FSNet uses a joint architecture to connect the semantic segmentation and corresponding failure detection network and trains these networks end-to-end using the semantic segmentation dataset. FSNet also allows the failure detection network to access internal features of the segmentation network during training and inference.

Let $S$ be a basic semantic segmentation network combining a convolutional encoder $S_E$ and decoder $S_D$. $S$ classifies each pixel of a given image $x$ of shape $w \times h \times 3$ into a particular label from a set $\mathcal{C} = \{1, 2, \ldots, C\}$. $S_D$ uses the convolutional features $e$ from the last layer of $S_E$ to generate logits $l$ of size $w \times h \times C$. Here $e = S_E(x)$ and $l = S_D(e)$. Based on architectural choice, $S_D$ may exploit features from different layers of $S_E$. Later, a softmax function is applied on $l$ to generate the predicted label map $\hat{y} = S(x) \in \mathcal{C}^{w \times h}$.

We are proposing a failure detection network $F$ to predict $\tilde{y}$, a 2D failure map of size $w \times h$ indicating the pixels where $\hat{y}$ is incorrect. $F$ consists of two encoders $F_{E1}$ and $F_{E2}$, and a decoder $F_D$. These encoders take $l$ as input after preprocessed by four different layers listed in Table I. First, $F_{E1}$ uses $x \oplus W_1(l)$ as its input and produces the encoded feature $f_1$. Meanwhile, $F_{E2}$ generates encoded feature $f_2$ from $W_2(l) \oplus W_3(l) \oplus W_4(l)$. Here, $\oplus$ represents channel-wise concatenation operation. Later, using Equation 1, $f_1$, $f_2$ and $e$ are concatenated to generate feature $z$ for the failure detection decoder $F_D$.

$$z = f_1 \oplus f_2 \oplus e.$$  \hspace{1cm} (1)

$F_D$ takes $z$ as input and upsamples it to generate the failure map $\tilde{y}$ of size $w \times h$. $\tilde{y}_{ij}$ represents the confidence of $F_D$ that $S$ has misclassified $x_{ij}$. Figure 2 shows an overview of FSNet and the inter-connection among its different components.

### B. Training Procedure

We use a single dataset and two different loss functions – segmentation loss and failure detection loss – to jointly train the semantic segmentation network $S$ and the failure detection network $F$ of FSNet. Let, for each input $x$, $S$ predict the label as $\hat{y}$. The cross-entropy loss function in Equation 2 is used to calculate the segmentation loss $L_1$ from ground-truth $\bar{y}$ and label prediction $\hat{y}$.

$$L_1 = -\sum_i \bar{y}_i \log(\hat{y}_i).$$  \hspace{1cm} (2)

The task of $F$ is to predict the failure of $S$. Therefore, a ground truth $\bar{y}$ is required to represent that failure, so that we can train $F$. Here, $\bar{y}$ is a 2D binary label of size $w \times h$. Each pixel of $\bar{y}$ is either one or zero, indicating the difference or similarity between $y$ and $\hat{y}$. We apply Equation 3 to generate this failure detection ground-truth $\bar{y}$. During training, $F$ is optimized to predict the $\bar{y}$. Assuming $\bar{y}$ as the output of the failure detection network, we use the balanced binary cross-entropy loss of Equation 4 to calculate the failure detection loss $L_2$ where $\beta$ is the weight balancing factor.

$$\bar{y}_{ij} = \begin{cases} 1 & \text{if } y_{ij} \neq \hat{y}_{ij} \\ 0 & \text{otherwise} \end{cases}$$  \hspace{1cm} (3)

$$L_2 = -\beta \cdot \bar{y} \log(\bar{y}) + (1 - \bar{y}) \log(1 - \bar{y})$$  \hspace{1cm} (4)

There are two steps in the training procedure. At first, we backpropagate only the loss $L_1$ into FSNet until the segmentation network $S$ is converged. This step only trains $S$ to perform semantic segmentation. After the convergence of $S$, $L_1 + L_2$ is used as the new loss. As $S$ and $F$ are connected, this step jointly optimizes semantic segmentation and failure detection networks. Our experiment shows that without converging $S$ first, we cannot jointly optimize both networks of FSNet.

| Layer | Operation | Description |
|-------|-----------|-------------|
| $W_1(l)$ | Conv1x1(l) | Applies 2D convolution on $l$ with kernel size 1. |
| $W_2(l)$ | Max(Softmax(l)) | Extracts maximum softmax value across each channel of $l$. |
| $W_3(l)$ | Sigmoid(Entropy(l)) | Uses sigmoid normalization after getting channelwise entropy of $l$. |
| $W_4(l)$ | ArgMax(l) | Returns indices of the maximum value across each channel of $l$. Same as segmentation label. |
TABLE II: Semantic segmentation accuracy in mIOU and Cls-Acc metrics. It shows the performance of individually trained SynthCP and jointly training FSNet segmentation model performance in three different datasets. Cityscapes dataset is used to train both approaches.

| Dataset      | Methods     | FCN8 mIOU | FCN8 Cls-Acc | DeepLabV2 mIOU | DeepLabV2 Cls-Acc |
|--------------|-------------|-----------|--------------|----------------|-------------------|
| Cityscapes   | SynthCP     | 29.92     | 39.23        | 32.98          | 37.36             |
|              | FSNet       | 29.84     | 35.19        | 33.22          | 41.70             |
| BD100k       | SynthCP     | 19.42     | 24.52        | 23.97          | 28.46             |
|              | FSNet       | 19.55     | 25.95        | 23.54          | 30.25             |
| Mapillary    | SynthCP     | 18.90     | 25.28        | 23.40          | 29.05             |
|              | FSNet       | 19.32     | 26.97        | 23.01          | 30.76             |

IV. EXPERIMENTAL SETUP

This section describes the experimental setup used for FSNet evaluation. First, we will discuss the in- and out-distribution dataset settings. Later, existing approaches, evaluation metrics, and implementation will be detailed.

**In- and Out-Distribution Dataset.** We used a training dataset of 2974 images from the Cityscapes [39] dataset to train FSNet and other approaches. There are two settings in the experimental setup. The first one is the in-distribution, where the training and testing data come from the same distribution. We used a testing dataset of 500 images from the Cityscapes for evaluation. In the out-distribution setting, the testing data comes from a different dataset. We used 1000 images from the BDD100k [40] and randomly selected 1000 images from the Mapillary [41] semantic segmentation dataset. All the segmentation networks are trained to segment the 19 classes available in the Cityscapes dataset.

**Methods to Compare.** We compare FSNet failure detection network to multiple methods – MSP [29], MC-Dropout [38], TCP [20], Direct-prediction [17] and SynthCP [16]. MSP and MC-Dropout are standard baselines for pixel-level failure detection. Direct-prediction, TCP, and SynthCP use a separate failure-dataset to train the failure detection network. Using this new dataset Direct-prediction uses a separate network to train their failure detector. TCP trains a network to predict the true class probability as a failure indicator. Most recently, SynthCP used a conditional GAN and a comparison module to train a model that identifies the failure of semantic segmentation. SynthCP is the SOTA approach among these works.

In contrast to existing approaches, FSNet jointly trains the semantic segmentation and the corresponding failure detection network. Although jointly trained, FSNet segmentation network should perform similarly to the individually trained segmentation network. To ensure this, we will compare FSNet segmentation network accuracy with the individually trained SynthCP segmentation network.

**Evaluation Metrics.** Following [16], [20], we used AP-Err, AP-Suc, AUC, and FPR95 as evaluation metrics. AP-Err considers incorrect prediction as positive class and computes the Area Under the Precision-Recall (AUPR) curve. AP-Suc computes AUPR too but considers correct prediction as to the positive class. AUC calculates the area under the Receiver Operating Characteristics, and FPR95 computes False-Positive Rate at 95% True-Positive Rate. FSNet segmentation network is compared with the SynthCP segmentation network using mean Intersection over Union (mIOU) and per-class accuracy (Cls-Acc). mIOU first calculates the IOU for each class and then calculates the average over classes. Cls-Acc measures the percentage of correctly labeled pixels for each semantic class and then averages over the classes.

**Implementation.** Following [16], we used FCN8 [42] and DeepLabV2 [43] as the semantic segmentation networks in our framework. FCN8 and DeeplabV2 are based on VGG16 and ResNet101 backbone networks, respectively, and pre-trained on the MS-COCO semantic segmentation dataset. Both encoders in the FSNet failure detection network use the ResNet18 network pretrained on the ImageNet dataset. Our training process consists of two steps. At first, the FSNet segmentation network is trained only using the segmentation loss for 20k iterations for convergence. For this step, we follow the hyper-parameters and image augmentations used by SynthCP. Then, for the next 10k iterations, FSNet is trained using segmentation and failure detection loss. The failure detection network uses adam optimizer with learning rate 0.0002.

V. EVALUATION

This section evaluates the semantic segmentation and failure detection accuracy of FSNet with the existing approaches. It also shows comparative performance for in-distribution and out-distribution settings.

A. Semantic Segmentation Evaluation

Table II shows the semantic segmentation accuracy between SynthCP and FSNet. In the in-distribution setting, the segmentation accuracy of FSNet based on FCN8 and DeepLabV2 improves by 4.0% in the Cls-Acc metric. FSNet also shows better performance in the mIOU metric. Both SynthCP and FSNet demonstrated lower accuracy than the in-distribution settings as BDD100k, and Mapillary datasets were unknown to the segmentation network. However, FSNet segmentation accuracy is better than SynthCP for out-distribution setting too. This result shows that we can train segmentation and failure detection networks jointly without degrading the segmentation accuracy.

B. Failure Detection Evaluation

Table III shows the failure detection accuracy of FSNet and all existing approaches using AP-Err, AP-Suc, AUC, and FPR95 metrics. These metrics are averaged over 19 classes of the Cityscapes dataset. For the in-distribution setting, FSNet failure detection network achieves 67.83% in AP-Err for identifying failure of FCN8, which is 12.30% higher than the SOTA – SynthCP. It also outperforms SynthCP in AUC and FPR95. However, FSNet is 0.2% inferior to SynthCP in the AP-Suc metric. In contrast to SynthCP, FSNet uses a balanced binary cross-entropy loss function to train the failure detection network. It has improved FSNet AP-Err by a large margin for a negligible performance reduction.
TABLE III: Failure detection experiments on the Cityscapes, BDD100k, and Mapillary dataset using AP-Err, AP-Suc, AUC, and FPR95 metrics. All approaches are trained using the Cityscapes training dataset to detect the failure of FCN8 and DeepLabV2 semantic segmentation networks. Test dataset Cityscapes refers to in-distribution and BDD100k, Mapillary refers to out-distribution settings.

| Test Dataset      | Methods              | FCN8                | DeepLabV2            |
|-------------------|----------------------|---------------------|----------------------|
|                   |                      | AP-Err↑ AP-Suc↑ AUC↑ FPR95↓ | AP-Err↑ AP-Suc↑ AUC↑ FPR95↓ |
| Cityscapes        | SynthCP [16]         | 55.53 99.18 92.92 22.47 | 49.99 99.34 92.98 21.69 |
|                   | MCDropout [38]       | 49.23 99.02 91.47 25.16 | 47.85 99.23 92.19 24.68 |
| in-distribution   | MSP [29]             | 50.31 99.01 91.54 25.34 | 48.46 99.24 92.26 24.41 |
|                   | Direct [17]          | 52.16 99.14 92.55 22.34 | 48.76 99.34 92.94 21.56 |
|                   | TCP [20]             | 48.54 98.82 90.29 32.20 | 45.57 98.84 89.14 36.98 |
|                   | FSNet (Ours)         | 67.83 98.98 94.35 21.52 | 57.84 99.13 93.97 21.39 |
| BDD100k           | SynthCP [16]         | 62.83 84.44 76.38 64.21 | 53.60 88.46 75.71 61.35 |
| out-distribution  | MCDropout [38]       | 59.77 83.78 74.85 64.68 | 38.26 84.18 66.27 70.10 |
|                   | MSP [29]             | 60.40 83.65 75.04 65.14 | 54.74 88.72 76.32 63.06 |
|                   | Direct [17]          | 61.77 84.91 76.85 62.51 | 53.18 88.28 75.53 61.81 |
|                   | TCP [20]             | 57.99 77.54 70.77 75.66 | 47.9 83.02 67.47 75.10 |
|                   | FSNet (Ours)         | 72.29 86.63 83.17 51.81 | 71.81 91.76 86.94 43.56 |
| Mapillary         | SynthCP [16]         | 58.64 92.97 82.22 46.91 | 49.51 93.72 79.57 47.05 |
| out-distribution  | MCDropout [38]       | 57.34 93.30 82.11 46.81 | 47.39 93.49 78.48 52.67 |
|                   | MSP [29]             | 58.29 92.97 82.53 45.35 | 48.03 93.65 79.32 46.63 |
|                   | Direct [17]          | 55.50 90.90 79.30 53.44 | 43.79 92.57 75.03 63.33 |
|                   | TCP [20]             | 56.21 93.23 81.77 47.19 | 32.97 92.88 73.61 55.43 |
|                   | FSNet (Ours)         | 68.29 93.76 87.75 40.19 | 63.19 95.68 88.95 37.36 |

Fig. 3: The Risk-Coverage curve for semantic segmentation failure detection. (a), (b) and (c) illustrate the Risk-Coverage curve for FCN8 failure detection in Cityscapes, BDD100k, and Mapillary datasets. (d), (e) and (f) demonstrate the Risk-Coverage curve for DeepLabV2 in the same dataset settings. Here (a) and (d) represent the in-distribution and (b), (c), (e), and (f) represent the out-distribution setting. In all cases, FSNet has a lower risk of failure for all coverage levels.

in the AP-Suc. In the same settings, FSNet for DeepLabV2 demonstrates a similar trend by outperforming SynthCP in the AP-Err metric by 7.85%.

Table III also shows FSNet failure detection accuracy for FCN8 and DeepLabV2 in out-distribution setting. Here, we trained FSNet using the Cityscapes dataset and evaluated using the BDD100k and Mapillary datasets. These experiments illustrate the FSNet generalization capability. In four
metrics and two datasets, \( FSNet \) outperforms all the existing methods.

We see higher AP-Err in out-distribution than in the in-distribution setting. The reason is the lower semantic segmentation accuracy for out-distribution (see Table II). As the segmentation networks make more label prediction errors in the out-distribution setting, the failure detection network can identify these errors and hence shows better performance in out-distribution than in-distribution.

C. Risk-Coverage Evaluation

We use the Risk-Coverage [30] curve to evaluate the impact after detecting semantic segmentation failure. Here, Coverage is the percentage of predicted pixel labels that are not flagged as a failure by \( FSNet \), and Risk is the percentage of misclassification error in those predictions. Based on this metric, \( FSNet \) can reject the segmentation network’s prediction to achieve the desired risk level.

Figure 3a shows Risk-Coverage curves for all approaches while detecting the failure of FCN8 on Cityscapes dataset for in-distribution setting. We plot these curves using ten different Coverage levels. Here \( FSNet \) demonstrates lower risk than all existing approaches. As an example, for 60% coverage, 10.5% risk means \( FSNet \) has rejected 40% segmentation prediction assuming that the prediction is incorrect, keeping the coverage 60%. In this coverage, 10.5% pixels have been incorrectly classified. All other methods show risk levels from 15% to 17.5%. For DeepLabV2 and in-distribution setting, \( FSNet \) risk of failure is 12% for 60% coverage while other existing approach’s risk varies from 15% to 16.5%.

Figure 3b and Figure 3c show risk-coverage curves for FCN8 in out-distributions setting. In both cases, \( FSNet \) show lower risk level than all existing methods for all coverage levels. Figure 3d and Figure 3f show risk-coverage curve for DeepLabV2 in out-distribution settings with the similar trend where \( FSNet \) outperforms all existing approaches.

Figure 4 shows qualitative results and the comparison between \( FSNet \) and SynthCP for detecting the failure of semantic segmentation.

D. Discussion

Based on experimental results, \( FSNet \) outperforms SynthCP and other existing approaches. In the ablation study, we experimented with multiple configurations to find critical components of \( FSNet \). These configurations include single and multi-branch architecture, full and partial-dataset. In a single-branch setting, we used only a single encoder to extract features from the input and logits of the segmentation network. In a multi-branch (see Figure 2), \( FSNet \) used two encoders to extract features from the input and logits output. We tested how the dataset size impacts \( FSNet \) using full-dataset and partial-dataset comprising 100% and randomly selected 20% Cityscapes training images to train \( FSNet \).

As described in the literature, SynthCP, Direct-prediction, and TCP train a segmentation network and apply that network on an unseen dataset to create a failure-dataset for failure detection training. Hence, the failure-dataset size is significantly smaller, which is 20% - 25% of the segmentation dataset. Therefore, these approaches can not take advantage of the entire available segmentation dataset. On the contrary, \( FSNet \) introduces a joint architecture and simultaneously uses the full semantic segmentation dataset to train
TABLE IV: We have trained a multi-branch failure detection network using the entire segmentation dataset and removed one feature at a time to study their impact on detecting the segmentation failure. To evaluate the effect of dataset size, we trained the multi-branch network consisting of all features using the partial-dataset. Besides, we trained a single-branch network to examine the performance difference between single and multi-branch failure detection networks in FSNet. Gray rows show the FSNet accuracy when the failure detection network is multi-branch and trained using the entire dataset.

| Dataset               | Multi Branch | Full Dataset | Removed Feature | FCN8 | DeepLabv2 | AP-Err↑ | AP-Suc↑ | AUC↑ | FPR95↓ | AP-Err↑ | AP-Suc↑ | AUC↑ | FPR95↓ |
|-----------------------|--------------|--------------|-----------------|------|-----------|--------|--------|-----|--------|--------|--------|-----|--------|
| Cityscapes in-distribution | ✓            | ✓            | x               | 60.01| 98.78     | 92.83  | 24.37  |      |        | 52.17  | 99.08  | 93.17| 22.85  |
|                       | ✓            | ✓            | W₁(l)           | 51.40| 98.04     | 89.37  | 39.42  |      |        | 45.53  | 98.34  | 89.49| 42.44  |
|                       | ✓            | ✓            | W₂(l)           | 61.88| 98.73     | 92.69  | 25.24  |      |        | 50.51  | 99.00  | 92.55| 24.63  |
|                       | ✓            | ✓            | W₃(l)           | 67.20| 98.96     | 94.17  | 21.99  |      |        | 53.19  | 99.08  | 93.25| 23.24  |
|                       | ✓            | ✓            | W₄(l)           | 45.85| 96.11     | 79.85  | 62.31  |      |        | 36.19  | 96.50  | 76.67| 64.90  |
|                       | ✓            | ✓            | S_E(x)          | 52.77| 98.54     | 91.26  | 26.75  |      |        | 54.78  | 99.13  | 93.65| 22.39  |
| BDD100k out-distribution | ✓            | ✓            | x               | 67.83| 98.98     | 94.35  | 21.52  |      |        | 57.84  | 99.13  | 93.97| 21.39  |
|                       | ✓            | ✓            | W₁(l)           | 65.92| 98.99     | 94.37  | 20.29  |      |        | 54.96  | 99.14  | 93.63| 22.63  |
|                       | ✓            | ✓            | W₂(l)           | 65.48| 98.90     | 93.83  | 23.59  |      |        | 50.64  | 99.03  | 92.70| 26.05  |
| Mapillary out-distribution | ✓            | ✓            | x               | 67.75| 85.36     | 80.59  | 55.15  |      |        | 50.09  | 91.74  | 86.71| 43.83  |
|                       | ✓            | ✓            | W₁(l)           | 62.38| 82.01     | 75.55  | 64.44  |      |        | 67.10  | 91.10  | 85.18| 51.26  |
|                       | ✓            | ✓            | W₂(l)           | 67.02| 85.39     | 80.74  | 54.43  |      |        | 67.86  | 91.38  | 85.73| 45.48  |
|                       | ✓            | ✓            | W₃(l)           | 71.81| 86.38     | 82.83  | 52.73  |      |        | 69.96  | 91.15  | 86.18| 47.32  |
|                       | ✓            | ✓            | W₄(l)           | 65.37| 85.24     | 79.09  | 58.70  |      |        | 65.03  | 90.58  | 82.80| 56.50  |
|                       | ✓            | ✓            | S_E(x)          | 64.27| 83.80     | 78.25  | 59.23  |      |        | 70.76  | 91.85  | 87.01| 43.68  |
|                       | ✓            | ✓            | x               | 72.29| 86.63     | 83.17  | 51.81  |      |        | 71.81  | 91.76  | 86.94| 43.56  |
|                       | ✓            | ✓            | W₁(l)           | 69.63| 85.27     | 81.12  | 55.51  |      |        | 68.14  | 90.74  | 85.32| 48.29  |
|                       | ✓            | ✓            | W₂(l)           | 72.00| 85.67     | 83.05  | 52.95  |      |        | 66.67  | 90.67  | 84.87| 49.92  |
|                       | ✓            | ✓            | x               | 65.08| 93.75     | 87.12  | 39.00  |      |        | 39.45  | 95.40  | 87.66| 40.34  |
|                       | ✓            | ✓            | W₁(l)           | 59.57| 91.25     | 82.68  | 51.93  |      |        | 56.46  | 95.87  | 87.41| 41.71  |
|                       | ✓            | ✓            | W₂(l)           | 59.86| 92.68     | 84.92  | 43.41  |      |        | 56.26  | 95.16  | 86.60| 43.24  |
|                       | ✓            | ✓            | W₃(l)           | 67.68| 93.78     | 87.57  | 40.60  |      |        | 58.61  | 94.97  | 86.84| 44.04  |
|                       | ✓            | ✓            | W₄(l)           | 61.18| 93.30     | 84.13  | 48.67  |      |        | 53.69  | 95.78  | 84.90| 47.48  |
|                       | ✓            | ✓            | S_E(x)          | 61.57| 92.82     | 85.35  | 42.41  |      |        | 59.84  | 95.43  | 87.77| 40.77  |
|                       | ✓            | ✓            | x               | 68.29| 93.76     | 87.75  | 40.19  |      |        | 63.19  | 95.68  | 88.95| 37.36  |
|                       | ✓            | ✓            | W₁(l)           | 66.22| 92.99     | 86.83  | 41.37  |      |        | 58.70  | 95.38  | 87.45| 41.52  |
|                       | ✓            | ✓            | W₂(l)           | 68.03| 92.62     | 86.53  | 44.08  |      |        | 54.35  | 94.37  | 84.89| 49.02  |

In all existing approaches, either semantic segmentation or a separate network is used for failure detection. However, FSNet failure detection network exploits multiple features from the segmentation network for failure detection. Figure 2 shows how these features are extracted from the segmentation network. We have removed one feature at a time from the FSNet multi-branch failure detection network to study the impact of different features. In Table IV, we have listed the feature name which is removed and the accuracy of FSNet in all metrics after removing them. It shows that the FSNet accuracy drops when any feature is removed from the failure detection network. Based on the ablation study, the most significant features are W₁(l), W₄(l) and S_E(x). Without these features, FSNet failure detection accuracy drops below the SynthCP. Table IV shows that our proposed joint architecture significantly improves the accuracy of FSNet for detecting the failure of semantic segmentation network.

VI. CONCLUSION

As deep learning-based semantic segmentation becomes an essential component for autonomous vehicles, identifying the segmentation failure is paramount for ensuring safety and robustness. This paper proposes a novel joint learning framework to train a semantic segmentation and corresponding failure detection network simultaneously. This failure detection network can identify the image area at pixel-level where the segmentation network has made an incorrect prediction. Therefore, our novel framework can inform downstream components in autonomous vehicle systems about expected semantic segmentation reliability. We show the effectiveness of FSNet using multiple datasets, segmentation networks, and evaluation metrics.

REFERENCES

[1] M. Treml, J. Arjona-Medina, T. Unterthiner, R. Durgesh, F. Friedmann, P. Schuberth, A. Mayr, M. Heusel, M. Hofmarcher, M. Widrich, et al., “Speeding up Semantic Segmentation for Autonomous Driving,” in MLITS. NIPS Workshop, vol. 2, no. 7, 2016.

[2] D. Feng, C. Haase-Schuetz, L. Rosenbaum, H. Hertlein, F. Duffhauss, C. Gläser, W. Wiesbeck, and K. Dietmayer, “Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges,” IEEE Transactions on Intelligent Transportation Systems, vol. 22, pp. 1341–1360, 2021.
