POLITECNICO DI TORINO
Repository ISTITUZIONALE

APSS Metrics for Fault Detection: Area, Position, Symmetry, and Shape in Image Segmentation

Original
APSS Metrics for Fault Detection: Area, Position, Symmetry, and Shape in Image Segmentation / Turco, V., Fezza, L., Ruospo, A., Sanchez, E., Sonza Reorda, M.. - ELETTRONICO. - (2025). (DFT 2025 38th IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems Barcellona (ESP) 21-23 October 2025) [10.1109/DFT66274.2025.11257474].

Availability:
This version is available at: 11583/3003629 since: 2025-10-05T12:20:14Z

Publisher:
IEEE

Published
DOI:10.1109/DFT66274.2025.11257474

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

20 November 2026

APSS Metrics for Fault Detection: Area, Position, Symmetry, and Shape in Image Segmentation

V. Turco, L. Fezza, A. Ruospo, E. Sanchez, M. Sonza Reorda
Politecnico di Torino, DAUIN, Torino, Italy

***Abstract*—In recent years, the impact of hardware-induced faults on neural networks performing image classification tasks has gained a lot of attention. Specifically, failures have been directly associated with wrong classifications. When it comes to different tasks, this association is less explicit. For example, the critical impact of hardware-induced faults on image segmentation tasks is less interpretable. In this work, we propose a novel technique for detecting critical permanent faults, relying on a dataset profiling phase to extract four metrics. These metrics are designed to assess and monitor the area, position, symmetry, and shape of prediction patterns across the output mask at the pixel level. Validation was performed through a statistical fault injection campaign on Fast-SCNN model trained on Cityscapes. To evaluate the effectiveness of the proposed method, a Faulty Output Dataset (FOD) was developed and employed to compare state-of-the-art (SOTA) metrics, such as Pixel Accuracy (PA) and mean Intersection over Union (mIoU), with the proposed one. The results show a high capability to detect critical faults, with an accuracy greater than 99%, comparable to SOTA methods, but with the advantage that the proposed method does not require a golden mask, increasing its applicability in real-world scenarios.**

***Index Terms*—Fault Injection, Reliability, Convolutional Neural Network, Bit-Flip, Online Test.**

I. Introduction

The use of Deep Neural Networks (DNNs) is steadily increasing in the deployment of complex and large-scale systems. Because of their inherent nature, the manifestation of hardware-induced faults can have highly variable effects: the impact may be negligible, tolerable, or, in the worst case, severely destructive. This depends on multiple factors, including the nature of the task, and the characteristics of the fault itself. Predicting the effects of hardware-induced faults on DNN outputs is crucial to the early identification of potential failures [1], [2]. Many state-of-the-art (SOTA) works are validated on image classification DNNs [3], [4], where it is relatively straightforward to determine whether the impact of a fault is critical: it is when it produces a wrong classification. For more complex tasks, such as image segmentation, the process of identifying and evaluating the effects of a fault is significantly more complicated. This is due to the presence of multiple predictions (one for each pixel) and the lack of a clear and shared definition of what *critical* means. As a result, it is difficult to make direct comparisons between the different approaches for fault detection/mitigation proposed in the literature.

For example, the authors in [5] proposed a novel fault detection technique for image segmentation models, using Average Precision 50 (AP50) as a criterion for defining a critical fault. In contrast, other works, such as [6] and [7], propose alternative criteria based on metrics such as mean Intersection over Union (mIoU) and Pixel Accuracy (PA).

This paper presents two main contributions. The first is the creation of a Faulty Output Dataset (FOD) (open-sourced [8]), generated using the Fast-SCNN model [9] and a subset of the Cityscapes validation set [10]. This dataset enables more accurate and consistent benchmarking of fault detection techniques as it includes 68k carefully annotated faulty masks, each labeled as either critical or tolerable according to well-defined criteria. The dataset was constructed through a statistical fault injection campaign on the Fast-SCNN model, followed by manual verification of each output mask according to a defined set of annotation rules. The second contribution is a fault detection technique for image segmentation DNN models. The method proposes the APSS metric, which leverages 4 fault-free measurements corresponding to Area, Position, Symmetry, and perimeter Shape (APSS) associated with the segmented classes. An initial calibration phase is used to derive optimal thresholds for these metrics, which are then applied during deployment to monitor any anomalous behavior. The proposed technique was validated using Fast-SCNN in conjunction with FOD, achieving over 99% coverage of critical faults. Furthermore, FOD enabled a comprehensive comparison with existing methods, experimentally demonstrating the performance of the proposed metrics. The paper is organized as follows. Section II reviews related work, providing context, and identifying the motivations for this study. Section III describes the proposed approach in detail. Section IV introduces the case study used to validate the method. Section V presents the experimental results and analysis. Finally, Section VI concludes the paper by summarizing the main contributions of the research work.

II. Related Work

In recent years, significant efforts have been dedicated to developing innovative fault detection and mitigation techniques aimed at improving the reliability of DNN models. Recent studies show that fault detection based on outlier analysis, such as out-of-range activations in model layers or values generated outside predefined thresholds, has proven effective in confining errors at both neuron and weight levels ([7], [11], [12]). In particular, the study [5], has explored the use of the distribution of neuron activations generated during model inference. This work focuses on profiling the activation distributions of fault-

free models, modeling a specific activation distribution for each layer. During inference, relying on the profiled distributions, an anomaly score is calculated by combining three metrics: the deviance of individual activations, the distance between the aggregate distribution of activations, and the variations in the "extreme" (max/min) values of neuron activations. Through this approach, silent data corruption (SDC) faults are injected into the system, detected with an average detection rate of 95%, and then mitigation techniques are applied. However, a key feature of the cited approaches is their reliance on access to the internal states of the model, thereby precluding their application in scenarios where the model must be treated as a black box. Moreover, when using DNNs for image segmentation tasks, experimental validation requires determining whether a SDC fault is critical for each inference. According to [5], a faulty inference is considered critical when the Average Precision 50 (AP50) score between the golden and the faulty output falls below the 0.5 threshold. Nonetheless, other criteria have been proposed in the literature to determine when an inference error should be considered critical.

To the best of our knowledge, only a few works in the literature have proposed alternative inference labeling rules to identify critical output masks resulting from hardware faults in image segmentation tasks. These rules are based on well-known evaluation metrics, namely Mean Intersection over Union (mIoU) and Pixel Accuracy (PA). A first study [6] proposes a taxonomy based on ΔmIoU, defined as the percentage difference between the mIoU of the faulty output and the fault-free (golden) output. The authors define four distinct fault labels:

- **Masked**: $\Delta mIoU = 0\%$
- **Accepted**: $0\% < \Delta mIoU \leq 5\%$
- **Warning**: $5\% < \Delta mIoU \leq 10\%$
- **Critical**: $\Delta mIoU > 10\%$

In contrast, in [7] the authors decide to use PA as a reference metric, defining three different types of fault labeling:

- **No impact SDC**: Pixel predictions between faulty and golden masks are identical
- **Tolerable SDC**: A fault modifies less than 1% of the total pixels *and* no classes vanish or appear
- **Critical SDC**: A fault modifies 1% or more of the total pixels *or* classes vanish or appear.

III. Proposed Approach

This research proposes a novel method for in-field identification and evaluation of criticalities during the deployment of image segmentation DNNs affected by hardware faults, relying solely on black-box access to the model. To the best of our knowledge, this specific aspect, within the context of DNNs for image segmentation, has not been thoroughly investigated. Previous studies have attempted to categorize and identify critical inferences and faults by leveraging metrics such as PA and mIoU, or by estimating the number of corrupted pixels or classes that can be tolerated before a fault becomes critical. However, these approaches present significant limitations. First, fault classification is often based on taxonomies that may introduce subjective bias. Second, many existing methods require access to fault-free network outputs to compare them with those affected by faults, an assumption that may not hold in real-world deployments. This work proposes a combination of four distinct metrics applied to the output masks of segmentation networks, enabling the extraction of multiple features such as **Area**, **Position**, **Symmetry**, and **Shape** (APSS) of the predicted classes. During a training phase on a specific dataset, these metrics are used to characterize the expected behavior of each feature and establish boundary thresholds. When these boundaries are exceeded during inference, the corresponding output is flagged as anomalous and potentially safety-critical.

Therefore, this methodology can be structured into two main phases:

1) **Threshold Calibration**: In this stage, the proposed APSS metrics are computed on the output masks of a fault-free DNN using a representative dataset. The distributions of these features are analyzed to define statistical thresholds (e.g., confidence intervals) that characterize the normal behavior of each feature under nominal conditions.
2) **Fault detection**: Once thresholds are established, APSS metrics are applied during inference to evaluate the output of the DNN under potentially faulty conditions. If any feature exceeds its corresponding threshold, the output is flagged as anomalous.

A. Metrics

Before detailing the approach used for the threshold definition and fault detection phases, it is important to define the four proposed metrics. The APSS metrics are: *Area*, *Position*, *Symmetry*, and *Shape (Right Angles)*.

1) *Area*

The area metric measures the surface occupied by a given class c within a segmentation output mask. It is computed as the total number of pixels labeled with that class:

$$A_c = \sum_{p \in M} \delta(p, c) \tag{1}$$

where:

- A_c is the total area (in pixels) for the predicted class c
- $\delta(p, c) = 1$ if pixel p is assigned to class c, and 0 otherwise.
- M is the segmentation mask.

This metric captures the extent of each class. It may be sensitive to intraclass variability, but it represents a computationally efficient indicator of abnormal predictions when class area distributions are expected to be stable.

2) *Position*

The position metric captures the spatial distribution of a predicted class within the segmentation mask by weighting pixel locations according to a Position Weight Matrix (PWM). The metric is computed as the sum of the element-wise product between the class mask and the corresponding PWM:

$$P_c = \sum_{(i,j)\in M} M_c(i,j) \cdot \mathrm{PWM}_c(i,j) \tag{2}$$

where:

- P_c is the resulting position for class c.
- $M_c(i,j) = 1$ if pixel (i,j) is predicted as class c, and 0 otherwise.
- $\mathrm{PWM}_c(i,j)$ is the positional weight associated with the pixel (i,j) for class c.

We exploited two different types of PWMs, each with different trade-offs:

- **Incremental PWM (Fig.1)**: a deterministic matrix where the weights increase progressively across the image.
- **Probabilistic PWM (Fig.2)**: a per-class matrix in which each pixel is assigned a probability representing the likelihood of that class appearing or not appearing at that location. These probabilities can be obtained empirically from clean segmentation outputs or defined heuristically based on prior knowledge (a given dataset).

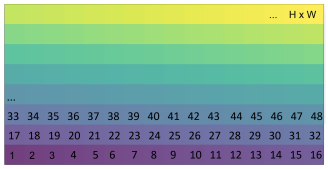


Fig. 1. Incremental matrix

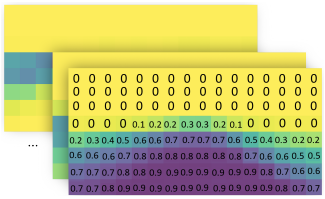


Fig. 2. Probabilistic class matrices

3) *Symmetry*

The symmetry metric evaluates the spatial symmetry of a predicted class in the segmentation mask. It measures how well one-half of the class mask aligns with the mirrored version of the other half, either horizontally or vertically. To compute this, the binary mask of the class is split horizontally or vertically, and one half is flipped and logically ANDed with the other. The resulting overlap is then weighted using a Symmetry Weight Matrix (SWM), and the final score is obtained by summing the weighted values:

$$S_c = \sum_{(i,j)\in H_c} B_c(i,j) \cdot \mathrm{SWM}_c(i,j) \tag{3}$$

where:

- S_c is the symmetry score fir the class c.
- B_c is the binary matrix resulting from the logical AND between one-half of the class mask and the opposite flipped half.
- H_c is the set of pixel indices in the half-mask.
- $SWM_c(i,j)$ is the weight of the SWM for class c.

As with the position, for symmetry we can adopt either *incremental* or *probabilistic* SWMs, each offering a trade-off between computational efficiency and spatial precision.

4) *Shape (Right Angles)*

The shape metric approximates the regularity of the predicted class contours by estimating the number of right angles formed by groups of connected pixels. This is based on the observation that segmentation networks often produce objects with shapes exhibiting a certain degree of geometric consistency, especially in man-made environments. To compute this metric, a pseudo-contour is first extracted by scanning the segmentation mask and identifying transitions between a class and its surroundings. Formally, the pseudo-contour mask $M \in \{0,1\}^{H\times W}$ is defined as:

$$\forall (i,j) \in [0, H-1] \times [0, W-1],$$

$$\mathrm{M(i,j)} = \begin{cases} 1 & \text{if } P(i,j) \neq P(i+1,j) \vee P(i,j) \neq P(i,j+1) \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where :

- $P(i,j)$ represents the pixel value in the predicted class mask at position (i,j).
- H and W represent the height and width of the segmentation mask.

Next, the metric proceeds in three steps:

1) Identify horizontal and vertical segments of at least n consecutive active pixels in M, producing two separate masks M_H and M_V
2) Compute their intersection $M_r = M_H \wedge M_V$, which corresponds to approximate right-angle corners.
3) The shape score R_c is given by the sum of active pixels in M_r:

$$R_c = \sum_{(i,j)} M_R(i,j) \tag{5}$$

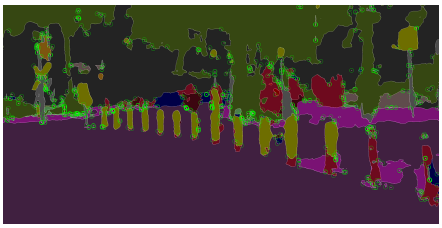


Fig. 3. Right Angles detection

Fig. 3 is a graphic description of which corners are identified by the proceedings, highlighted by the green circles. The final value R_c, which is the sum of all the green circles, represents an approximation of the number of right angles in the predicted class region. The sensitivity of this metric depends on the parameter n, which defines the minimum segment length used to detect a corner.

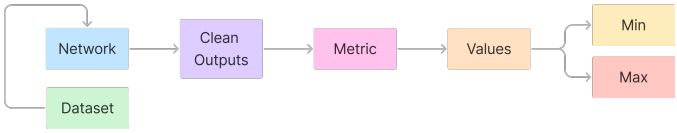


Fig. 4. Threshold decision process

B. Threshold Calibration and Fault Detection

The process for identifying the presence of a fault through the execution of an inference is based on the use of **thresholds** that define whether a value among the metrics is anomalous or not. Lower bounds and upper bounds are defined for each of the metrics, depending on the type of metric or class analyzed, which define the 'normal' behavior of the model. When reference data or prior knowledge are available, thresholds can be defined directly based on this information. However, in more complex scenarios where such data are not accessible, the simplest approach is based on a statistical strategy. In this case, as depicted in the Fig. 4, thresholds are calculated by analyzing a large and representative set of images in order to capture more variability in the input scenes. The analysis must be performed on a specific reference dataset, chosen to reflect an appropriate variety of scenarios expected during deployment. As a result, the technique may be dataset-dependent. This dataset serves as the basis for determining the standard values of the metrics. The process of calculating the lower and upper bounds is based on the metric values derived from the clean outputs obtained from a fault-free network processing the image set.

The approach is based on the execution of all four key metrics, each of which has its own thresholds. *A fault is detected if at least one of these metrics identifies an anomaly based on its respective threshold boundaries.* Specifically, the Area metric evaluates the proportion of pixels occupied by a particular class and checks whether this value falls outside the established bounds. The Right Angles metric focuses on detecting angular irregularities along the contours of segmented classes, identifying areas where right angles are formed. Position and Symmetry metrics, on the other hand, analyze the spatial properties of the predicted classes by comparing pixel alignment and symmetry across the mask, either using incremental or probabilistic weight matrices (SWM and PWM) to refine detection.

IV. Case of Study

The experiments were conducted on a popular image segmentation network, Fast-SCNN [9], trained using the Cityscapes [10] dataset. Cityscapes is a widely utilized resource for image segmentation and urban scene understanding. The dataset contains 5,000 high-quality images, divided into three partitions: training, validation, and test sets. It includes 30 different classes, although only 19 of them are effectively available for training, while the remaining, much rarer classes, are grouped into an empty class (identified as -1). The model obtains a performance of 54.84% in terms of mIoU and 92.37% for PA over the Cityscapes validation partition. The experiments have been executed on a GPU GeForce RTX 3060 Ti, while the framework used is PyTorch, which is compatible with the Fault Injector (FI) used [13].

To evaluate the performance of the fault detection metrics, a dataset was created by injecting permanent faults into the synaptic weights of the Fast-SCNN network. This dataset, called *Faulty Output Dataset* (FOD) and provided open-source in [8], was generated using a subset of the Cityscapes validation set, which is known for its high variability in urban scenes. For each of the 500 images in the validation set, statistical fault injection was performed within the model by generating a fault list with 2% error and 99% confidence, corresponding to 8,439 faults. These faults were introduced by modifying specific weights in the Fast-SCNN model, flipping the most significant bit in a randomly selected synaptic weight, thus significantly altering the parameter values due to the FP32 representation. Therefore, a total of 68,001 random image-fault pairs were selected for the FOD, from the 8439×500 faulty outputs, representing a great variety of fault injections and their corresponding output labels. This approach was chosen because it maximizes the likelihood of generating critical faults while keeping the fault injection process manageable.

The FOD dataset features a pronounced class prevalence, with 88.10% of the entries labeled as critical and 11.90% as non-critical. Additionally, a significant portion of the dataset (43.07%) consists of masks where all pixels belong to a single class, most commonly 'road'. The remaining 56.93% comprises more complex masks with varying class distributions. The FOD dataset includes information on fault parameters, such as the frame index and the fault details (including the affected layer, tensor index, and bit position), as well as performance metrics such as mIoU and PA. However, these metrics are not used for labeling the faults. Instead, the labels are manually assigned based on whether the output mask is classified as critical (C) or non-critical (NC). To determine whether a mask is considered critical, we defined three heuristic rules. If at least one of these points is satisfied, the mask is labeled as critical. The three rules are as follows:

- Uniform Class Labeling (Fig. 6): The mask consists of pixels that are all assigned to the same class, ensuring that every pixel in the mask is consistently labeled.
- Large Spatially Continuous Misclassified Regions (Fig. 7): The mask contains large, spatially continuous areas of misclassified pixels that disrupt the visual coherence of the segmented object, resulting in a distortion of its perceived shape.
- Incorrect Classification of Specific Categories (Fig. 8): The mask includes contiguous regions of pixels from categories that involve the presence of a human being, spanning from class 11 (person) to class 18 (bicycle), which are incorrectly classified.

This custom FOD dataset serves as a foundation for evaluating the effectiveness of the proposed fault detection approach.



Fig. 5. Clean image



Fig. 6. Uniform Class Labeling

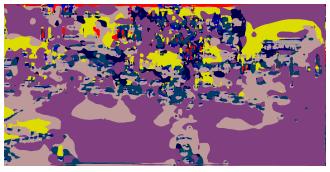


Fig. 7. Large Spatially Continuous Misclassified Regions

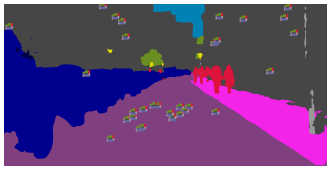


Fig. 8. Incorrect Classification of Specific Categories

V. Experimental Results

This section presents an analysis of all the results obtained by comparing and describing the various configurations involving the proposed metrics, using both clean masks and FOD data. Each component is first evaluated individually, considering multiple possible configurations to show their specific contributions and costs. Then, from each unit, the final metric is progressively built. Furthermore, the achieved performance is evaluated alongside SOTA methods to assess differences, vulnerabilities, advantages, and effectiveness of the proposed approach.

A. Thresholds analysis

This step is necessary because each metric, in order to be operational in the fault detection phase, needs a set-up process as described in III-B. This process is concerned with analyzing data and statistics to define useful thresholds to identify faults. It is important to note that the FOD dataset, derived from the Cityscapes validation dataset, is never used for any parameter selection or threshold computation. To perform these preliminary analyses, the only data used are the clean fault-free output of the DNN, obtained from the Cityscapes training partitions. Moreover, two different threshold estimation strategies were adopted: one based on the training partition, which involves known data, and another based on the test partition, which represents unseen data. Since the training partition consistently outperformed the test-based one across all experiments in terms of overall performance, all the results reported in the following will refer exclusively to the training-based strategy.

Tables I and II collect all thresholds obtained from the analyses of the 4 developed metrics. The first table presents all upper and lower thresholds for the area, position, and symmetry metrics, computed for each class in the cityscapes dataset. The second table, on the other hand, defines the thresholds for the number of right angles, which apply uniformly to all output masks. For this metric, a value of n = 3 was selected, as higher values would lead to an exponential loss of information, significantly reducing both the accuracy and efficiency of the metric.

TABLE I
Summary of threshold statistics (min, max) for each class across the metrics.

Class	Area		Position		Vert. Sym.		Horiz. Sym.	
	min	max	min	max	min	max	min	max
road	1309.0	1.2e6	6.4e8	7.6e11	0.0	1.7e11	0.0	4.8e10
sidewalk	0.0	8.1e5	0.0	4.4e11	0.0	8.1e10	0.0	8.3e9
building	0.0	1.6e6	0.0	1.9e12	0.0	4.6e11	0.0	3.1e11
wall	0.0	9.0e5	0.0	1.5e12	0.0	3.4e11	0.0	1.9e10
fence	0.0	6.7e5	0.0	9.6e11	0.0	1.2e11	0.0	3.6e10
pole	0.0	1.5e5	0.0	2.1e11	0.0	1.5e10	0.0	8.8e9
traffic light	0.0	4.2e4	0.0	7.3e10	0.0	8.0e9	0.0	0.0
traffic sign	0.0	1.4e5	0.0	2.3e11	0.0	1.0e10	0.0	7.9e9
vegetation	0.0	1.2e6	0.0	1.8e12	0.0	4.0e11	0.0	1.2e11
terrain	0.0	5.6e5	0.0	2.9e11	0.0	4.7e10	0.0	9.0e9
sky	0.0	5.3e5	0.0	9.3e11	0.0	1.8e11	0.0	5.4e8
person	0.0	7.4e5	0.0	7.7e11	0.0	1.3e11	0.0	6.8e10
rider	0.0	7.7e4	0.0	8.8e10	0.0	1.0e10	0.0	2.1e9
car	0.0	8.4e5	0.0	8.8e11	0.0	1.5e11	0.0	9.4e10
truck	0.0	7.0e5	0.0	1.0e12	0.0	9.4e10	0.0	2.5e10
bus	0.0	5.6e5	0.0	7.2e11	0.0	6.0e10	0.0	3.7e10
train	0.0	1.3e6	0.0	1.8e12	0.0	4.3e11	0.0	8.3e10
motorcycle	0.0	2.1e5	0.0	2.0e11	0.0	2.2e10	0.0	4.2e9
bicycle	0.0	2.9e5	0.0	2.7e11	0.0	4.4e10	0.0	9.4e9

TABLE II
Right Angles threshold statistics (min, max)

Right Angles	Min	Max
n=3	13.0	1220.0

B. Fault detection results

At this stage, the thresholds are used to identify inferences affected by critical and non-critical faults in the FOD dataset. Table III presents the results for each APSS metric in the form of confusion matrices. The table reports the following four terms, defined as:

- **True Positive (TP)**: Inferences affected by a critical fault that are correctly classified as critical.
- **False Negative (FN)**: Inferences affected by a critical fault that are incorrectly classified as tolerable.
- **True Negative (TN)**: Inferences not affected by a critical fault that are correctly classified as tolerable.
- **False Positive (FP)**: Inferences not affected by a critical fault that are incorrectly classified as critical.

TABLE III
APSS Metrics Coverage Results

Metric	TP	FN	TN	FP
Area	96.37%	3.63%	99.79%	0.26%
Position	95.76%	4.24%	99.74%	0.21%
Symmetry	97.74%	2.26%	98.83%	1.17%
Shape	56.76%	43.24%	99.80%	0.20%

The results of the first metric (i.e., the **area**), are shown in the first table row. This metric already achieves a high rate of TNs, identifying 99.79% of tolerable inferences correctly, as well as a high rate of TPs, identifying 96.37% of critical inferences. Consequently, the goal of the next metrics is to further reduce the number of FNs, harmful to the system, by reclassifying them as TPs. At the same time, the aim is to minimize the FPs rate, avoiding the unnecessary rejection of tolerable inferences for the correct system's operation. As for

the **position** and **symmetry** metrics, both were evaluated using incremental and probabilistic weight matrices. In both cases, the incremental matrix yielded comparable or superior results to the probabilistic one. Moreover, given its lower deployment cost, only the results obtained with the Incremental setup are reported in the second and third rows of the table. The position metric shows a strong ability to classify a high number of tolerable inferences, achieving a TN rate of 99.79% on its own. Conversely, the symmetry metric excels in identifying critical inferences, reaching a TP rate of 97.74%. Finally, the analysis of right angles, in the last row of the table, differs significantly from the other three metrics. It exhibits a particularly high number of FNs; however, this does not imply that the metric is not useful. On the contrary, it may capture within the TPs some inferences not identified by the other three metrics, while maintaining the lowest FPs rate, around 0.2%.

TABLE IV
APSS AND SOTA COVERAGE RESULTS

Metric	TP	FN	TN	FP
Proposed APSS	99.05	0.95	98.17	1.83
SOTA PA	99.97	0.03	88.92	11.08
SOTA mIoU	99.40	0.60	99.81	0.19

To enable comparison with classification methods from the SOTA, the four metrics are combined into a single approach, the APSS. The general idea is to label as critical any output mask that has been deemed to be not tolerable by at least one of the four metrics included in the proposed method. The desired effect of merging the four metrics is to achieve superior critical fault identification performance while maintaining an acceptable FPs rate for good computational efficiency. The final results obtained using the APSS metrics are presented in Table IV, along with the results evaluated using PA and mIoU, for comparison purposes. The combination of the metrics achieves a coverage of output masks affected by critical faults greater than 99%, with a cost of discarded inferences of only 1.83% of all tolerable cases in the FOD dataset. These results are very promising, especially when compared to those obtained with current SOTA approaches. Regarding PA, the classification of critical masks reaches a very high value; however, it also incurs a significant error, discarding more than 11% of fault-free inferences. In contrast, mIoU produces results more similar to those achieved by the proposed method. However, both PA and mIoU share a major limitation: they rely on a direct comparison between the faulty mask and the golden mask, which is not available in real-world scenarios.

VI. CONCLUSIONS

This paper proposes a new technique, called APSS, for fault detection in DNNs for image segmentation. The technique is based on the use of four distinct metrics designed to extract and evaluate the characteristics of output masks. To enable accurate evaluation and objective comparison with the SOTA, a custom FOD was created and made publicly available to facilitate the reproducibility of the experiments. The experimental results show that the combination of the four metrics achieves high coverage of inferences affected by critical faults, while maintaining a low erroneous termination rate of inferences affected by tolerable faults.

VII. ACKNOWLEDGMENT

This work was supported by project SERICS (PE00000014) under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU. This study was carried out within the FAIR - Future Artificial Intelligence Research and received funding from the European Union Next-GenerationEU (PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR) – MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3 – D.D. 1555 11/10/2022, PE00000013). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

REFERENCES

[1] Y. He, M. Hutton, S. Chan, R. De Gruijl, R. Govindaraju, N. Patil, and Y. Li, "Understanding and mitigating hardware failures in deep learning training systems," in *Proceedings of the 50th Annual International Symposium on Computer Architecture*, ser. ISCA '23. New York, NY, USA: Association for Computing Machinery, 2023. [Online]. Available: https://doi.org/10.1145/3579371.3589105

[2] V. Turco, A. Ruospo, E. Sanchez, and M. Sonza Reorda, "Early detection of permanent faults in dnns through the application of tensor-related metrics," in *2024 27th International Symposium on Design Diagnostics of Electronic Circuits Systems (DDECS)*, 2024, pp. 13–18.

[3] S. K. S. Hari, M. B. Sullivan, T. Tsai, and S. W. Keckler, "Making convolutions resilient via algorithm-based error detection techniques," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 4, pp. 2546–2558, 2022.

[4] E. Ozen, O. Ozerdem, and A. Orailoglu, "Linear algorithmic checksums for deep-neural-network error detection: Fundamentals and recent advancements," *IEEE Design Test*, vol. 42, no. 3, pp. 26–40, 2025.

[5] D. Ma *et al.*, "Dr. dna: Combating silent data corruptions in deep learning using distribution of neuron activations," in *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3*, ser. ASPLOS '24. New York, NY, USA: Association for Computing Machinery, 2024, p. 239–252. [Online]. Available: https://doi.org/10.1145/3620666.3651349

[6] G. Gavarini, A. Ruospo, and E. Sanchez, "A fast reliability analysis of image segmentation neural networks exploiting statistical fault injections," in *2023 IEEE 24th Latin American Test Symposium (LATS)*, 2023, pp. 1–6.

[7] S. Burel, A. Evans, and L. Anghel, "Techniques for detecting and masking faults in semantic segmentation applications," *Microelectronics Reliability*, vol. 157, p. 115397, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0026271424000775

[8] L. Fezza *et al.*, "Fod: Feature oriented detection," https://github.com/cad-polito-it/fod, 2024, accessed: 2025-05-12.

[9] R. P. K. Poudel, S. Liwicki, and R. Cipolla, "Fast-scnn: Fast semantic segmentation network," 2019. [Online]. Available: https://arxiv.org/abs/1902.04502

[10] M. Cordts *et al.*, "The cityscapes dataset for semantic urban scene understanding," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 3213–3223.

[11] Z. Chen, G. Li, and K. Pattabiraman, "A low-cost fault corrector for deep neural networks through range restriction," in *2021 51st Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2021, pp. 1–13.

[12] F. Geissler *et al.*, "Towards a safety case for hardware fault tolerance in convolutional neural networks using activation range supervision," 2021. [Online]. Available: https://arxiv.org/abs/2108.07019

[13] V. Turco *et al.*, "SFIadvancedmodels ," https://github.com/cad-polito-it/SFIadvancedmodels, Accessed: 2024-10-10.