Hardware faults that matter: Understanding and Estimating the safety impact of hardware faults on object detection DNNs

Syed Qutub[1], Florian Geissler[1], Yang Peng[1], Ralf Gräfe[1], Michael Paulitsch[1], Gereon Hinz[2], and Alois Knoll[2]

[1] Dependability Research Lab, Intel Labs, Germany
syed.qutub@intel.com
[2] Technical University of Munich, Germany

Abstract Object detection neural network models need to perform reliably in highly dynamic and safety-critical environments like automated driving or robotics. Therefore, it is paramount to verify the robustness of the detection under unexpected hardware faults like soft errors that can impact a system's perception module. Standard metrics based on average precision produce model vulnerability estimates at the object level rather than at an image level. As we show in this paper, this does not provide an intuitive or representative indicator of the safety-related impact of silent data corruption caused by bit flips in the underlying memory but can lead to an over- or underestimation of typical fault-induced hazards. With an eye towards safety-related real-time applications, we propose a new metric **IVMOD** (Image-wise Vulnerability Metric for Object Detection) to quantify vulnerability based on an incorrect image-wise object detection due to false positive (FPs) or false negative (FNs) objects, combined with a severity analysis. The evaluation of several representative object detection models shows that even a single bit flip can lead to a severe silent data corruption event with potentially critical safety implications, with e.g., up to $\gg 100$ FPs generated, or up to $\sim 90\%$ of true positives (TPs) are lost in an image. Furthermore, with a single stuck-at-1 fault, an entire sequence of images can be affected, causing temporally persistent ghost detections that can be mistaken for actual objects (covering up to $\sim 83\%$ of the image). Furthermore, actual objects in the scene are continuously missed (up to $\sim 64\%$ of TPs are lost). Our work establishes a detailed understanding of the safety-related vulnerability of such critical workloads against hardware faults.

1 Introduction

Research communities seek to make the deployment of general artificial intelligence (AI) and deep neural networks (DNNs) used in everyday life as dependable as possible. Significant emphasis is placed on handling corrupted input (e.g. due to visual artifacts or to attacks) provided to the model. However, less effort has been dedicated to studying corruptions of the internal state of the model itself, most importantly caused by faults in the underlying hardware. Such faults can occur naturally, such as memory corruption induced by external (e.g., cosmic neutron) radiation or electric leaking in the circuitry itself, typically manifested as bit flips or stuck-at-0/1s in the memory elements [1, 16], which may alter the DNN model parameters (*weight faults*) or the intermediate states (*neuron faults*). Platform faults can also impact the input while it is held in memory, yet this work focuses on the computational part of the DNN as our goal is to estimate the vulnerability of the model. The impact of these faults is often unpredictable in systems with large complexity. Alterations can be of transient or permanent nature: Transient faults have a short life span of the order of a few clock cycles and are therefore harder to detect by the system. On the contrary, permanent faults may silently corrupt the system output for a longer period. Memory protection techniques like error correcting code (ECC) can mitigate the risk of hardware faults [20]; however, they are typically applied only to selected elements to avoid significant cost overheads. Given the rise in technology scaling with smaller node sizes and larger memory areas, future platforms are expected to become even more vulnerable to hardware faults [20].

Object detection DNNs are among the most common examples of highly safety-critical DNN applications as they are in autonomous vehicles or in medical image analysis. Typically, autonomous systems process events based on perception techniques. Hence, it is critically important that any potential hazards does not impact the system-level evaluation of events. While the chances for a

(a) Inference from Yolov3 model and Kitti dataset

(b) Inference from Faster-RCNN model and Kitti dataset

Figure 1: Examples of the impact of a single neuron bit flip (at bit position b and layer index l, see image insets). TPs are marked by green, FPs by red and FNs by blue rectangles, comparing the fault-free (top) and the faulty (bottom) predictions. In example (a) multiple FPs are generated right in front of the ego vehicle, while in (b) all previous detections are erased due to the fault.

hardware fault to occur (for example, the chance of a neutron radiation event hitting a memory element) can be estimated statistically, it remains unclear how to quantify the safety-related impact of the failure of a DNN applied for the purpose of object detection. In contrast to simpler classification problems, the model output here typically consists of a multitude of bounding boxes and classes per image, of which a subset can be altered in the presence of a fault while others remain intact, see Fig. 1. We find that commonly used average precision (AP) [19] metrics inappropriately rely on the count of false objects irrespective of their interrelations (grouping in the same image or distributed across multiple frames). In real-time applications of DNNs, it further matters if the corrupted output is volatile or temporally stable across multiple input frames. The user is typically behind a tracking module that can regularize instantaneous alterations. We, therefore, see the need

to establish a safety-related assessment of the vulnerability of object detection workloads under soft errors. Depending on the specifications from safety assessment, we adopt a generalized notion of a safety hazard as a perturbation that causes a potentially unsafe decision by the end-user of the object detection module.

Therefore, we introduce two variants of the metric IVMOD (Image-wise Vulnerability Metric for Object Detection), namely $\text{IVMOD}_{\text{SDC}}$ in case of an image-wise silent data corruption (SDC), and $\text{IVMOD}_{\text{DUE}}$ in case of detectable uncorrectable errors (DUE)s.

In this paper, we discuss the characteristics of the AP-based metrics in detail when used to quantify a model's vulnerability. For example, AP50 is found to be hypersensitive to rare single corruption events compared to an evaluation at the image level. Our work supports maintaining the relationship of the system-level hazard evaluation to the impact of any hardware faults. We find that a hardware fault - if it hits the crucial bits of either neuron or weight - can silently lead to excessive amounts of additional false positives (FPs) and increase the rate of false negatives (FNs) misses. We further study the impact of permanent faults in a real-time situation by considering continuous video sequences and observing a significant frequency that the error manifestation persists for a critical time interval.

In summary, this paper makes the following contributions:
- We demonstrate that AP-based metrics lead to misleading vulnerability estimates for object detection DNN models (Sec. 4.1)
- We propose an SCD-based/DUE-based metric IVMOD to quantify the vulnerability of object detection DNN models under hardware faults (Sec. 4.2).
- We evaluate the vulnerability of various representative object detection DNN models using the proposed IVMOD, illustrating the probability of a single bit flip resulting in a potentially safety-critical event (Sec. 5.1).
- For each such event, we propose various quantitative metrics to estimate the impact severity for typical safety-critical applications (Sec. 5.2).
- We extend our image-based evaluation to a video-based safety-critical system and measure the vulnerability of temporal persistency ($A_{\text{FP}_{\text{blob}}}$ and $A_{\text{FN}_{\text{blob}}}$) due to a permanent fault, by tracking the FPs and FNs across multiple video frames (Sec. 6).

2 Related Work

The effort to estimate the vulnerability or resilience of the DNNs against hardware faults affecting the model has been explored recently to study the safety criticality of a model when used in real-time operation.

To this extent, faults are injected in DNNs during inference either at the application layer on weights/neurons ([6, 16]), or by neutron beam experiments (([4, 9]), black-box techniques). Authors of Ref. [2, 16] considered transient faults, which are multiple event upsets occurring in data or buffers of DNN accelerators. Many prior works claimed DNNs to have inherent tolerance towards faults. Li. et al. ([2]) studied the vulnerability of DNNs by injecting faults in data paths and buffers with different data type levels and quantified it in the form of SDC probabilities and FIT (failure in time) rates. It is seen that errors in buffers propagate to multiple locations compared to errors in data-path. These works estimated the resiliency of the model by injecting multiple fault injections during the feed-forward inference. This analysis is limited to image classification models like AlexNet [14], VGG [24], and ResNets [7]. Our analysis does not characterize the faults in buffer and faults in the data-path. We assume the faults will propagate to the application layer, which may impact either the weights or the neurons. Hence we analyze them independently, assuming equal probabilities. The Ares framework [21] demonstrated that activations (neurons) in image classification networks are 50x more resilient than weights. These works focus mainly on fault models involving multiple bit flips captured by bit error rate (BER). There is limited research done on understanding the vulnerability of object detection DNNs. The work in Ref. [4] quantified the architectural vulnerability factor (AVF) of Yolov3 using metrics like SDC AVF, DUE AVF, and FIT rates. This work studies fault propagation by injecting a random value in the selected register file and not flipping a bit. The authors argue that not all SDCs are critical, given that change in objects' confidence scores after injecting faults is tolerable. The definition of SDC used in this work is not straightforward. They use the precision and recall values computed at the object level by combining all the images, obscuring the actual vulnerability. The vulnerability of object detection DNNs is studied by injecting faults using neutron beam [18]. The authors analyzed both transient

and permanent faults but not on continuous video sequences. Also, the dataset considered in these experiments was primarily limited to only one object per image. Also, they injected faults into the input image. We limit our fault injections to neurons and weights and only to convolution layers of the DNNs as the fully connected layers did not change much of the observed data. We believe fault-injected images do not fall into the category of the model vulnerability. They rather find their place in adversarial input space within various adverse fault/noise models. The results obtained from many of these works are not easy to compare as the failure and SDC definitions differ and do not follow standard baseline. To our best knowledge, our paper is the first to demonstrate vulnerabilities of the object detection models in detail using the proposed (**IVMOD**) metrics to measure the severities at the image level. Also, we introduce a new metrics $A_{\mathrm{FP_{blob}}}$ and $A_{\mathrm{FN_{blob}}}$ quantifying the area occupancy of FP/FN blobs, which is essential to establish the safety criticality of the object detection models concerning specific real-time applications.

3   Preliminaries

3.1   Hardware faults vocabulary

Our fault injection technique includes transient and permanent faults. Transient faults refer to random bit flips (0→1 or 1→0) of a randomly chosen bit, which occur during a single image inference and are removed afterward. Permanent faults are modeled as stuck-at-0 and stuck-at-1 errors, meaning that a bit remains consistently in state '0' or '1' without reacting on intended updates. Those faults are assumed to persist across many image inferences. We inject faults either into intermediate computational states of the network (neurons) or into the parameters (weights) of the DNN model, focusing only on convolutional layers, which constitutes a significant part of all operations in the studied DNNs. Both types of faults represent bit flip in the respective memory elements, holding either temporary states such as intermediate network layer outputs or learned and statically stored network parameters. A fault can potentially induce critical alterations of the model predictions, measured by $\mathrm{IVMOD_{SDC}}$ or $\mathrm{IVMOD_{DUE}}$ as shown in Eq. 1.

3.2   Experimental setup: Models, datasets and system

We use standard object detection models - Yolov3 [22], RetinaNet [17], Faster-RCNN (F-RCNN[23]) - together with the test datasets CoCo2017 [19], Kitti [5] and Lyft [12]. We retrained Yolov3 on the Kitti and Lyft dataset, and the Faster-RCNN model on Kitti for comparative experiments. We used open-source trained weights for the rest of the models and datasets. The base performances of these models in terms of AP50 and mAP can be found in Fig. 3. The parameter configurations used for these models (NMS threshold, confidence score, etc.) are taken from the original publications. Since fault injection is compute-intensive, we select a subset of 1000 images for each dataset to perform the transient fault analysis and use a single Lyft sequence of 126 images for the permanent fault analysis. All experiments adopt a single-precision floating-point format (FP32) according to the IEEE754 standard [10]. Our conclusions also apply to other floating-point formats with the same number of exponent bits, such as BF16 [11], since no relevant effect was observed from fault injections in mantissa bits.

4   Methodology of vulnerability estimation

4.1   Issues with average precision

In object detection, evaluation and benchmarking methods are most commonly selected from the family of average precision (AP)-based metrics (in combination with specific IoU thresholds such as AP50 or mAP). Libraries such as CoCo API [19] perform the following relevant steps to obtain AP values from a set of object predictions: i) ground truth and the predicted objects are collected in groups of the same class label, ii) within a group, the predicted objects are sorted w.r.t their confidence scores, iii) the sorted predictions are consecutively assigned to the ground truth objects within the same class group, using an appropriate IoU threshold, iv) precision and recall (PR) curves are evaluated sequentially through the confidence-ranked TP, FP, and FN objects, v) the class-wise AP is calculated as the area under the interpolated PR curve of a class, and vi) the overall AP is determined as the average of the class-wise AP values.

It has been pointed out that such AP metrics can lead to non-intuitive results in the detection performance of a model on a specific data set [22]. In the following, we illustrate that an AP-based

Figure 2: Simulation of the effect of fault injection on the AP metric. Here, an artificial data set of 100 objects was generated, where each object was classified as TP with a chance of 0.7 or as a FN otherwise. In addition, FPs were created with a rate of 0.3 per true detection. Both TPs and FPs are assigned random confidence values between 0.7 and 1. To this setup (a), additional FPs simulating the effect of fault injection were augmented or existing TPs were randomly eliminated to model fault-induced FNs ((b)-(d)). The diagrams show the PR curves and the effect of fault injection on them. Number and confidence range of the faulty objects are given in the insets.

evaluation can be misleading when estimating the vulnerability of a model against corruption events such as soft errors in a safety-critical real-time context concerning the probability and severity of corruption. Corruption events lead to additional FP and FN objects merged into or eliminated from the healthy list of detected objects. We identified the following issues when trying to quantify model vulnerability based on AP metrics:

- **Object-level evaluation:** The AP is calculated on an object level, i.e., the amount of TP, FP, FN objects accumulated across all images is used for evaluation. This does not consider how corrupted boxes are distributed across images, i.e., one image with a large number of fault-induced FP detections can have the same effect as many corrupted images with few FP detections each. From a real-time safety perspective, however, the amount of corrupted image frames is typically relevant, as this may determine, for example, the robustness of a video stream used for environment perception.
- **Dependency of PR on confidence:** Due to the sequential and integration-based character-istic of the average precision, the fault-induced FP object's impact depends highly on those sample's confidence. This does not reflect the potential safety relevance a low-confidence FP object may have, see more below.
- **Dependency of box assignment on confidence:** The strict confidence ranking can, in some cases, lead to a non-optimal global assignment of bounding boxes. For example, a better matching box might have slightly lower confidence than a global optimization would demand.
- **Class-wise average:** Common and rare classes have the same weight in the overall AP metric. However, their detection performance and vulnerability can be quite different as they typically relate to the samples the model encountered during training.

In particular, the second point above is non-intuitive; we therefore illustrate this in more detail in Fig. 2 with the help of a generic example from a randomly generated data set of 100 objects. Additional FPs with low confidence compared to the reference set of objects have a negligible impact on the metric as they get appended to the tail of the PR curve, even when numerous and potentially safety-critical. On the contrary, few high-confidence FP objects can lead to significant drops in the AP as those samples get sorted in at the head of the PR curve to lower it. Fault-induced FNs reduce the area under the PR curve by pushing the samples towards smaller recalls.

4.2   Proposed metrics: IVMOD

We introduce IVMOD metrics to measure the image-wise vulnerability of the object detection DNNs. Our evaluation strategy described in the following seeks to counter the issues with AP-based metrics described in the last section in order to reflect vulnerability estimation better addressing safety targets. In particular, our approach is characterized by:

- **Image-level evaluation:** We evaluate vulnerability on an image level instead of an object level. This approach reflects that those faults jeopardize safety applications that silently alter the free and occupied space by inducing false detections in an image, particularly sequences

thereof, even if such an alteration involves only few false objects per frame. We register image-wise SDC and DUE events, see Sec. 4.2, to determine the probability of a relevant fault impact. The severity of the latter is evaluated separately in terms of the amount of induced FPs and FNs. Due to their image-based character, $\text{IVMOD}_{\text{SDC}}$ and $\text{IVMOD}_{\text{DUE}}$ metrics are naturally independent of the object confidences.

- **Confidence-independent box assignment:** False-positive objects can be critical whether they have high or low confidence, which is masked in the AP metric. We apply a different assignment scheme for FPs and FNs that omits confidence ranking and hence makes the model vulnerability metric independent of the confidence of FPs, see Sec. 4.2. The assignment strategy can also be varied to relax class correspondence requirements, which are often overemphasized from a safety perspective. The system can perform at degraded level if its sure of object location and not much about the class.
- **Class-independent average:** We evaluate the overall sample mean instead of the mean of individual class categories to reflect typical imbalances in the data set concerning object classes.

Assignment policy In contrast to the sequential and class-wise matching described in Sec. 4.1, we calculate the cost matrix from a set of predictions and ground truth objects for a single image. The cost for matching objects is the IoU between the bounding boxes. If the IoU is below the specified threshold $\text{IoU}_{\text{eval}} = 0.5$, or if the classes of the two objects are not the same, we assign a maximum cost. To analyze the relevance of exact class predictions for an application, we can harden or soften the class matching from a one-to-one correspondence to compatible class clusters or neglect class matching altogether. A Hungarian association algorithm [15] is then deployed to obtain the global optimal cost assignment. As usual, the number of accepted matches per image represents the true positive (TP) cases. False detections are registered in the following cases: i) a FP and a simultaneous FN detection occurs if the IoU with the assigned ground truth object is below the threshold, independent of the predicted class, or if the IoU is sufficiently large, but the classes are not compatible, ii) a single FP occurs if there is a predicted object that cannot be assigned to any ground truth object with acceptable costs, iii) a single FN occurs if there is a non-assigned ground truth object. Fig. 1 shows an illustrative example of assigned TP, FP, FN boxes. In our setup, we clip predicted bounding boxes reaching out of the image dimensions – e.g., due to faults – to the actual image boundaries.

IVMOD ($\text{IVMOD}_{\text{SDC}}$ and $\text{IVMOD}_{\text{DUE}}$) We define the $\text{IVMOD}_{\text{SDC}}$ rate as the ratio of events where a fault during inference causes a silent corruption of an image and the total number of image inferences. $\text{IVMOD}_{\text{SDC}}$ is an SDC defined as a change in either of the TP, FP, or FN count of the respective image, compared to the original fault-free prediction, given that no irregular *NaN* (not a number) or *Inf* (infinite) values occur during the inference as shown in Eq. 1. Since TPs and FNs are complementary to each other, we can eliminate either TP or FN in $\text{IVMOD}_{\text{SDC}}$ in Eq. 1. On the other hand, the $\text{IVMOD}_{\text{DUE}}$ rate is the ratio of events where irregular *NaN* or *Inf* values are generated during inference and detected inside the layers or in the predicted output due to the injected fault in the respective image during inference and are computed using the Eq. 1. As DUE events are naturally detectable, they typically are less critical than SDC events. Explicitly,

$$\text{IVMOD}_{\text{SDC}} = \frac{1}{N} \sum_{i=1}^{N} \left\{ \left[(FP_{\text{orig}})_i \neq (FP_{\text{corr}})_i \vee (FN_{\text{orig}})_i \neq (FN_{\text{corr}})_i \right] \wedge \neg \text{Inf}_i \wedge \neg \text{NaN} \right\},$$

$$\text{IVMOD}_{\text{DUE}} = \frac{1}{N} \sum_{i=1}^{N} \left[\text{Inf}_i \vee \text{NaN}_i \right]. \tag{1}$$

5   Transient faults

Our evaluation concept is guided by the assumption that in safety-critical applications, both the miss of any existing object as well as the creation of any false positive object can be potentially hazardous. Therefore, we consider the probability that such an SDC event occurs and our primary metrics $\text{IVMOD}_{\text{SDC}}$ and $\text{IVMOD}_{\text{DUE}}$ (Eq. 1) captures the vulnerability of a model. For transient faults, this evaluation is performed in Sec. 5.1. Accordingly, we independently inject 50,000 random single-bit flips in neurons and weights at each inference of the chosen test datasets. Subsequently, Sec. 5.2 discusses the severity of each of those SDC events in terms of the average impact of additional FP and FN objects, their size, and confidence. If a specific use case is given, the factors of probability and severity can be used to derive the risk of an error [13].

(a) Neuron faults

(b) Weight faults

Figure 3: Key metrics to interpret the vulnerability of object detection DNNs in the presence of transient hardware faults: (left) AP50, (center) mAP, (right) error rate, distinguishing $IVMOD_{SDC}$ and $IVMOD_{DUE}$. We study both neuron faults (a) and weight faults (b).

5.1   Corruption probability

In Fig. 3, we present the fault injection campaigns of all studied networks, comparing the typical benchmark metrics AP-50 and mAP to the $IVMOD_{SDC}$ and $IVMOD_{DUE}$ rate as defined in Sec. 4.2. Both Yolov3 and RetinaNet show a significant change in the AP-50 and mAP metrics under the injected neuron and weight faults: The accuracy can drop as much as from 89.4% to 34.4% (AP-50) due to a single weight fault in the scenario of Yolov3 and Kitti. On the other hand, F-RCNN does not showcase much sensitivity to the injected faults ($\lesssim 0.8\%$ change in AP-50). At the same time, the $IVMOD_{SDC}$ rates vary between 0.4% and 1.8% (neuron faults), and from 1.5% to 4.2% (weight faults). This discrepancy illustrates the need for a more realistic vulnerability estimate. As shown below, in Tab. 1, fault injections in Yolov3 and RetinaNet tend to produce many FPs with statistically increased confidence. This

Figure 4: Example of the AP50 PR curves of few classes from Yolov3 and Kitti in the fault free and faulty cases.

leads to a drastic shift of the PR curves, as shown in the example in Fig. 4, where only 3.2% out of 1000 samples have corrupted prediction (demonstrated the similar effect in Fig. 2(c)). Rare classes are susceptible to such faults, diminishing the class-averaged metric further. Since the induced false objects are concentrated on only a few images, the AP metric exaggerates the safety-related vulnerability of the model under software errors (see also the discussion in Sec. 4.1).

In contrast, the F-RCNN model architecture appears to be very robust against the generation of FPs (see Tab. 1). Predictions made in the presence of a soft error have nearly the same confidence as in the fault-free case. However, faults do disturb the detection of objects as a significant portion of FNs appear (on average between $10 - 33\%$). Nevertheless, the AP metrics for FRCNN under fault injection are hardly affected: We observe very few accuracies drops for both neurons and weights. At the same time about $0.4 - 0.7\%$ ($1.5 - 1.8\%$) of the images see silent data corruption. In this case, the AP-based metric is masking the potentially safety-critical impact of underlying faults. We further observe that for Yolov3, $IVMOD_{DUE}$ events are generated in $\sim 0.9\%$ of the neuron injection cases, while in RetinaNet and F-RCNN and for weight injections, those are negligible ($\lesssim 0.1\%$). We conclude that the Yolov3 architecture stimulates neuron values that have a higher chance if being flipped to a configuration encoded as *NaN* or *Inf* (in FP32, all exponential bits have to be in state '1'), compared to RetinaNet and F-RCNN. The weight values of all networks, on the other hand, are closely centered around zero, which makes it very unlikely to reach a *NaN*

or *Inf* bit configuration [6] (typically MSB and at least another exponential bit are in state '0' at the same time). We observe that the faults injected in weights at any bit of FP32 cause higher IVMOD$_{\mathrm{SDC}}$ rates than the faults injected at the neuron level. They showcase $\sim 2\times$ more adverse effects on predictions than faults injected at the neuron level.

5.2   Corruption severity

We next aimed to understand how faults leading to IVMOD$_{\mathrm{SDC}}$ events corrupt images and how the severity of an IVMOD$_{\mathrm{SDC}}$ event on a potential safety-critical application can be estimated. Even though the relevance of a safety feature may depend on the specific application, we identified the following fundamental features to serve as a specific indicative measure of an SDC fault severity, see Tab. 1:

- The average number of FP objects induced by a given IVMOD$_{\mathrm{SDC}}$ fault and the proportion of boxes lost due to a fault, referred to as ΔFP and ΔFN_{n}, respectively as described in Eq. 2 (subscript 'n' represents normalization as the upper limit of FNs is known, in contrast to FPs).
- The average size of objects in the presence and absence of SDC (avg(size)) since a significant change of the object size can be safety-critical,
- The average area of the image that is erroneously occupied due to IVMOD$_{\mathrm{SDC}}$ induced FP objects ($A_{\mathrm{FP_{blob}}}$) and the average portion of the vacant area created by not detecting the objects due to IVMOD$_{\mathrm{SDC}}$ faults ($A_{\mathrm{FN_{blob}}}$).
- The average confidence of objects in the presence and absence of IVMOD$_{\mathrm{SDC}}$, avg(conf).

We motivate this choice more in the following subsections.

Table 1: Severity features averaged over all IVMOD$_{\mathrm{SDC}}$ events.

	Yolo+Coco	**Yolo+Kitti**	**Yolo+Lyft**	**Retina+Coco**	**F-RCNN+Coco**	**F-RCNN+Kitti**
Neurons:						
avg(ΔFP)	**333**	36	174	33	0	0
avg(ΔFN_{n})(%)	42.2	41.3	**46.6**	16.1	25.3	33.3
avg(conf) (corr, orig)	0.99, 0.52	0.99, 0.51	0.99, 0.65	**0.79, 0.11**	0.73, 0.73	0.90, 0.89
avg(size)/$1e^3$px (corr, orig)	4.3, 11.2	**34.5, 2.3**	17.8, 7.3	5.6, 20.3	17.0, 18.6	6.3, 6.8
$A_{\mathrm{fp\text{-}occ}}$(%)	36.8	**62.5**	59.8	0.7	1.7	0.0
$A_{\mathrm{fn\text{-}vac}}$(%)	4.0	5.1	4.8	**53.1**	41.1	39.8
Weights:						
avg(ΔFP)	**198**	59	145	7	0	0
avg(ΔFN_{n})(%)	23.3	21.7	21.3	4.0	9.6	**29.6**
avg(conf) (corr, orig)	1.00, 0.53	1.00, 0.52	1.00, 0.65	**0.62, 0.11**	0.72, 0.73	0.89, 0.88
avg(size)/$1e^3$px (corr, orig)	5.5, 12.1	21.4, 2.5	**30.8, 6.9**	7.9, 19.8	10.0, 15.0	4.9, 5.0
$A_{\mathrm{fp\text{-}occ}}$(%)	40.1	**81.0**	79.1	1.5	0.3	0.0
$A_{\mathrm{fn\text{-}vac}}$(%)	15.1	2.5	6.8	42.3	77.8	**85.8**

Fault-induced object generation and loss Object detection is commonly used in scenarios where the number of objects, combined with their location and class, is input to safety-critical decision making. Examples include face detection or vehicle counting in traffic surveillance, automated driving, or medical object detection. Therefore, to assess IVMOD$_{\mathrm{SDC}}$ severity, we quantify the impact of a fault injection by the differences (a loss in TPs equals the gain in FNs)

$$\Delta FP = (FP_{\mathrm{corr}} - FP_{\mathrm{orig}}),$$
$$\Delta FN_{\mathrm{n}} = (TP_{\mathrm{orig}} - TP_{\mathrm{corr}})/TP_{\mathrm{orig}}, \tag{2}$$

In Tab. 1, we observe that all Yolov3 and RetinaNet scenarios exhibit large numbers of fault-induced FPs ($\gg 100$ in Yolov3 and Coco experiments). For neuron faults, the generation of FPs is, on average, more pronounced. Furthermore, the normalized FN rates show that already a single

fault can cause a significant loss of accurate positive detections. Average FN rates are higher for neuron faults than weight faults and reach averages up to 47% (Yolov3 and Lyft). F-RCNN models are robust against the generation of FP objects but not immune against fault-induced misses (e.g. Fig. 1b). The number of generated FPs and FNs varies in a broad sample range, up to the maximum limit of allowed detections (here 1000), due to the inhomogeneous impact of flips in different bit positions (see Sec. 5.3). In some situations, additional objects created by faults will match actual ground truth objects, leading to a negative FP or FN difference. This effect originates from the imperfect performance of the original fault-free model and is tolerated here due to the minor impact. By relaxing the class matching constraints from one-to-one correspondence to no class matching, we can further segment the type of FPs that the IVMOD$_{\text{SDC}}$ events cause. It appears that situations where an FP is due to a change in the class label only or due to a shift of the bounding box only (on average $\lesssim 3$ for Yolo models, 0 for others). In most cases, *both* the bounding box gets shifted, and the class labels is mixed up, or predicted objects cannot be matched with any ground truth object at all.

Object size and confidence Box sizes and confidence values are other severity indicators since large erroneous objects take up a more significant portion of the image space, and high-confidence objects might be handled with priority in some use cases. Tab. 1 shows the change of the average box size and confidence of all model detections across the identified IVMOD$_{\text{SDC}}$ events. In most models, the typical box size is reduced in the presence of faults, which is partially due to the creation of boxes with zero width or height. However, there are also scenarios where faults tend to induce overly large objects (Yolov3 and Kitti, Lyft, see Tab. 1) that can even fill the entire picture. An object's average confidence score after fault injection significantly increases in the scenario of Yolov3 and RetinaNet, while there is hardly any impact on F-RCNN predictions. This explains why confidence-sensitive metrics based on AP react differently to fault injections in the respective architectures; see the discussion in Sec. 4.1.

Area occupancy safety-related decision-making in a dynamic environment is most importantly based on the detected free and occupied space. For example, an automated vehicle will determine a driving path depending on the detected drivable space. A large number of false-positive objects, even when small in size, can, in combination, cover a significant portion of the image, which will leave only little free space. On the opposite, in some situations, they may overlay each other and occupy only a little space. To reflect a realistic severity of free space, we first cluster all FP and FN objects to *blobs* by projecting them to a binary space of occupancy and vacancy (see Fig. 6). As we are only interested in fault-induced false objects, our blobs for a given frame at time t are defined as follows:

$$FP_{\text{blob}} = \mathcal{I}(\text{det}_{\text{corr}} - \text{det}_{\text{orig}}),$$
$$FN_{\text{blob}} = \mathcal{I}(\text{det}_{\text{orig}} - \text{det}_{\text{corr}}). \tag{3}$$

$$A_{\text{FP}_{\text{blob}}} = |FP_{\text{blob}}|/A_{\text{image}},$$
$$A_{\text{FN}_{\text{blob}}} = |FN_{\text{blob}}|/|\mathcal{I}(\text{det}_{\text{orig}})|. \tag{4}$$

In Eq. 3, det denotes the set of all detected bounding boxes (TP and FP), and $\mathcal{I}(x)$ represents the pixel-wise projection to binary occupancy space, i.e., for any pixel u in a blob x it is $\mathcal{I}(u < 0) = 0$, $\mathcal{I}(u \geq 0) = 1$ (see Fig. 6). We define the occupancy coefficients in Eq. 4, where A_{image} is the size of the image in pixels and $|\dots|$ denotes the sum of all nonzero pixels in a blob. In Tab. 1, we see Yolo+Kitti creates significantly less ΔFP than Yolo+CoCo, but the average $A_{\text{FP}_{\text{blob}}}$, in this case, is $\sim 2x$ greater than $A_{\text{FP}_{\text{blob}}}$ of Yolo+CoCo. This can even be observed using the feature avg(size)/$1e^3$ (average size of bounding boxes of all the detections combined - TPs+FPs). In the case of Yolo+Kitti, the avg(size)/$1e^3$ is $15x$ and $\sim 8x$ larger than its original detections when a fault is injected in neurons and weights. This implies that ΔFP alone cannot determine the safety impact during an IVMOD$_{\text{SDC}}$ event. Similarly, F-RCNN creates no ΔFP, but large free space ΔFN_{n} by missing the TPs. F-RCNN+Kitti, when induced with weight faults, is more safety-critical as the $A_{\text{FN}_{\text{blob}}}$ is highest compared to other studied models. Furthermore, in case of neuron faults, the RetinaNet and F-RCNN have higher $A_{\text{FN}_{\text{blob}}}$.

5.3   Bit-wise analysis of false object count

The severity of an IVMOD$_{\text{SDC}}$ event typically depends on the magnitude of the altered values, where values with a considerable absolute value are more likely to propagate and disrupt the network predictions [6, 8, 16]. Therefore, the severity features are expected to form a non-uniform

(a) FP Neurons

(b) FP Weights

(c) FN Neurons

(d) FN Weights

Figure 5: Bit-wise analysis of the severity of IVMOD$_{SDC}$ events. Diagrams show the FP difference (a), (b) and FN rates (c), (d) for neurons and weights, respectively. Bit 31st is the sign bit, 30th bit being the most significant bit and 23rd bit is the lowest bit of exponent part.

distribution depending on the flipped bit position. To gain a better intuition, we here choose to present a bit-wise analysis of the ΔFP and ΔFN_n samples during the IVMOD$_{SDC}$ events. To quantify the impact of bits, we define the bit-averaged false-positive difference, bitavg(ΔFP), which intuitively tells us how many FPs an SDC event with a particular bit position induces, on average. Similarly, for FNs, the normalized bit-averaged difference, bitavg(ΔFN_n), represents what portion of the originally detected objects disappears due to an SDC event with a specific bit position. In Fig. 5, we observe that, for neuron faults, those additional FPs are typically caused by bitflips in either of the three highest exponential bits, as long as those do not lead to DUE instead. For weight faults, we find a situation similar to classifier networks where the specific value range of weights centered around zero is encoded in bit constellations where the MSB is in state '0' while the next higher exponential non-MSB bits are in state '1', see Ref. [6]. This explains why almost only MSB flips induce large values and IVMOD$_{SDC}$ (with a high number of FPs). Given the respective relevant neuron and weight bit flips, the ΔFN_n ratio is increased up to 90% (meaning that portion of all true positive detections is lost) in some of the models as shown in Fig. 5(c),(d). In particular, due to MSB and other high exponential bit flips the average bitavg(ΔFN_n) is $\sim 47\%$. We observe that FN alterations can, to some extent, be induced also by lower exponential bits.

(a) orig

(b) corr

(c) FP_{blob}

(d) FN_{blob}

Figure 6: Illustration of the clustering of bounding boxes to binary occupancy blobs. In this example we find from (c) and (d) that $A_{FP_{blob}} = 33.3\%$, $A_{FN_{blob}} = 7.5\%$ (white pixels indicate space occupied by fault FPs).

6 Permanent faults

Our analysis in this section aims to understand whether permanent stuck-at faults (see Sec. 3.1) leads to temporally consistent errors on an object level leading to continuous failure. The object

Figure 7: Pixel wise tracking of FP blobs. First row: orig dt are fault free detections. Second row: corr dt are faulty detections. Third row are tracked FP-blobs (white pixels are occupied by FP blobs).

Figure 8: Pixel wise tracking of FN blobs. First row: orig dt are fault free detections. Second row: corr dt are faulty detections. Third row are tracked FN-blobs (white pixels is the free space created by FNs).

detection model typically receives sequential images from a continuous video stream in real time applications. We assume a permanent hardware fault hitting the inference module which in turn causes persistent miss detections on consecutive images. In this case, they will appear either as ghost objects in the output (as FPs) or lead to a consecutive miss of an object (as FNs) - both situations can be highly safety-critical. A perception pipeline typically also includes a tracking module for detected objects, which can then be used to predict an object's trajectory and make an informed decision concerning the next maneuver of the vehicle. Therefore, we simulate a simple tracking of instantaneous fault-induced FPs and FNs clusters to determine whether they would be persistent in a realistic scenario. For the analysis in this section, we use Yolov3 and the Lyft data set. This is the only dataset used in our analysis that provides consecutive images from video sequences (Lyft sequence of the CAM_FRONT channel featuring 126 frames is considered). From our experiments with transient faults injections in Sec. 5, we understand that no effect is observed by altering mantissa bits or by flips in the direction $'1' \to \ '0'$ since this does not generate large

(a) Average area occupancy by FP-blob

(b) Average area occupancy by FN-blob

Figure 9: Tracking of FP- and FN-blob area

(a) FP

(b) FP Neurons

(c) FP Weights

(d) FN

(e) FN Neurons

(f) FN Weights

Figure 10: Vulnerability of Yolov3 and Lyft for permanent faults.

values. Therefore, the experiments of this section are accelerated by using only stuck-at-1 faults in the exponential bits of FP32. However, results have been rescaled to account for the probability of injections in all 32 bits. In this section, we designed an experiment where we inject each of 1000 single random permanent faults (exponential bits) at neurons and weights independently for the above considered sequence to understand its safety impact.

6.1 Evaluating fault persistence

We track the movement of blobs (Eq. 3) using a simple pixel-wise M/N tracking scheme [3]. The proposed tracker incorporates the following criteria to establish that a given pixel of FP or FN blob is persistent, at a given frame t: i) The pixel occupied in at least M/N consecutive frames. (if it is also occupied in the current frame, this corresponds to t track update; otherwise it is a coasting track), ii) If the occupancy of that pixel in the last N frames is below M, we check the vicinity around that pixel for past occupancy. Deploying a simplified unidirectional motion model, we register a persistent dynamic pixel for the current frame if occupancies above M are found in the past N frames in a close enough (here 50 pixel, abbr. px) vicinity.

For FN blobs, we omit coasting due to the nature of detection misses. After registering the persistent pixels computed by the pixel-wise tracker, the occupied ($A_{\mathrm{FP_{blob}}}$) or free-space ($A_{\mathrm{FN_{blob}}}$) area is calculated using Eq. 4. The tracking parameters are chosen as (10/15): The upper frame number is hereby estimated from a critical time of reaction to a persistent false target (≈ 0.5s) and the frame rate of the Lyft sequence (30Hz), leading to $N = 0.5\mathrm{s} \cdot 30\mathrm{s}^{-1} = 15$ key sequential frames. This estimated upper number can be application specific relevant to its safety specifications.

6.2 Corruption probability and severity

In Fig. 7 and Fig. 8, we show examples of persistent FP and FN blobs in selected frames. The occupied ($A_{\mathrm{FP_{blob}}}$) and free ($A_{\mathrm{FN_{blob}}}$) space of an entire video sequence is presented in Fig. 9a and

Fig. 9b. For orientation, we also give the area difference between original and ground truth predictions (Fig. 9), $A_{\text{FP}_{\text{ref\_blob}}} = |\mathcal{I}(\text{det}_{\text{orig}} - \text{gt})|/A_{\text{image}}$ and $A_{\text{FN}_{\text{ref\_blob}}} = |\mathcal{I}(\text{gt} - \text{det}_{\text{orig}})|/|\mathcal{I}(\text{det}_{\text{orig}})|$ (where gt is ground truth). We neglect these contributions originating from the model imperfection as it is a function of training and is found to be small (in the above examples $< 1\%$) compared to the fault-induced occupancy ($\sim 66\%$ and $\sim 62\%$, respectively). The example demonstrates that tracked FP blobs may persist across the entire image sequence and occupy a significant amount of free space. Similarly, a significant portion of the image can be lost persistently across the sequence (it reaches as high as $\sim 96\%$). Our statistical evaluation from 1000 permanent fault injections on the selected image sequence is given in Fig. 10 for FP and FN. The Fig. 10(a) and (d) shows both the SDC probability (in the form of persistant occurance) and the severity ((b)-(c), (e)-(f)) in detail. We register an SDC for a given fault if any persistent FP or FN is found during the sequence with a severity of at least level L. The severity L is quantified as the average area occupied by the blob (for FP normalized by the image size, for FN by the TP blob size, see above). The severity levels are varied from 0% to 15% in Fig. 10 to illustrate the effect of softening or hardening of the safety requirements. As the severity of a fault is again expected to depend on the bit position of the injected fault, we present both bit-selected and bit-averaged numbers in Fig. 10(b,c,e,f).

In this figure, the permanent faults in neurons and weights have a probability of 1.8% and 3% to create persistent ghost FP objects with a minimal area of $L > 0$, respectively. With $L > 15\%$ of an image area, this reduces to 0.9% and 2.9%, respectively. On average, faults hitting MSB bit in weights on this model have 96% probability to manifest into a persistent FP blob of area $> 81\%$. On the other hand, persistent FN blobs incorrectly indicating vacant spaces occur with a much lower chance. Bitflips cause persistent objects only in the highest exponential bits in case of neurons or in the MSB bits in the case of weights. This observation is consistent with the findings from transient faults in Sec. 5. Using the given area occupancy metrics, permanent weight faults have a higher severity than neuron faults; in particular, weight faults on average induce massive ghost FP blobs of $> 83\%$ of the image area.

7   Conclusion

This work points out the challenges in estimating the vulnerability of object detection models under bit flip faults. Average precision-based metrics are either very sensitive or not sensitive to the corruption events, which can be misleading in a safety context. For example, for F-RCNN+Kitti, neuron injections experiments showed almost no impact ($< 0.1\%$) in the AP50 and mAP metrics. Using the image-based evaluation metric IVMOD proposed here, however, we see that 0.7% of all images lose substantial amounts ($> 30\%$) of the total TP detections due to a single bit flip. The evaluation method presented in this work allows us to come to a vulnerability estimate better addressing safety targets. Given the IVMOD$_{\text{SDC}}$ probabilities and severities (see Fig. 3 and Tab. 1), we conclude that the chances of safety-related corruptions due to soft errors are minor to moderate ($0.4\% - 4.2\%$) in the studied setups. IVMOD$_{\text{SDC}}$ events due to weight faults are about two times as likely as neuron faults. However, if SDC occurs, the severity can be grave. The IVMOD metric should always be considered in combination with severity features for safety purposes. This is because IVMOD does not quantify the severity, but only considers the existence of false and missed bounding boxes. Our metric is defined relative to the original performance. This means that even if a fault also acts in a beneficial way, i.e. fixing some FP or FN occurrences, it will be categorized as a SDC here. We estimated this severity with the help of different safety-related features. We observed that high bits of the exponent of floating point numbers, when hit by either neuron or weight faults, can lead to a significant increase in ΔFP and ΔFN_{n}. This effect is also translated into an average occupancy value that reflects the area portion of the image that is critically altered by a fault. We find that large average occupancies (up to $A_{\text{FP}_{\text{blob}}} \sim 81\%$ for FP and $A_{\text{FN}_{\text{blob}}} \sim 86\%$ for FN) are common, reflecting significant safety hazards. Finally, we studied the use case of a sequential real-time image sequence from Lyft to show that permanent *stuck-at* faults on neurons or at weights can induce FP objects covering as much as $\sim 83\%$ of the image area, creating dangerous ghost objects. Similarly, up to $\sim 63\%$ of the TP area in the scene can be missed. Overall, the weight faults are more likely impactful than neuron faults and have a higher severity in area occupancy (except for permanent FNs).

References

1. Athavale, J., Baldovin, A., Graefe, R., Paulitsch, M., Rosales, R.: AI and Reliability Trends in Safety-Critical Autonomous Systems on Ground and Air. Proceedings - 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN-W 2020 (2020)
2. Beyer, M., Morozov, A., Valiev, E., Schorn, C., Gauerhof, L., Ding, K., Janschek, K.: Fault Injectors for TensorFlow: Evaluation of the Impact of Random Hardware Faults on Deep CNNs. 30th European Safety and Reliability Conference, ESREL 2020 and 15th Probabilistic Safety Assessment and Management Conference, PSAM 2020 (2020)
3. Blackman, S., Popoli, R.: Design and Analysis of Modern Tracking Systems (Artech House Radar Library). Artech house (1999)
4. Dos Santos, F.F., Navaux, P., Carro, L., Rech, P.: Impact of reduced precision in the reliability of deep neural networks for object detection. Proceedings of the European Test Workshop (2019)
5. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The KITTI dataset. International Journal of Robotics Research (2013)
6. Geissler, F., Qutub, S., Roychowdhury, S., Asgari, A., Peng, Y., Dhamasia, A., Graefe, R., Pattabiraman, K., Paulitsch, M.: Towards a safety case for hardware fault tolerance in convolutional neural networks using activation range supervision. In: Proceedings of the Workshop on Artificial Intelligence Safety 2021 co-located with the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI 2021), Virtual, August, 2021 (2021)
7. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (2016)
8. Hong, S., Frigo, P., Kaya, Y., Giuffrida, C., Dumitras, T.: Terminal brain damage: Exposing the graceless degradation in deep neural networks under hardware fault attacks. Proceedings of the 28th USENIX Security Symposium (2019)
9. Hou, X., Breier, J., Jap, D., Ma, L., Bhasin, S., Liu, Y.: Security Evaluation of Deep Neural Network Resistance against Laser Fault Injection. Proceedings of the International Symposium on the Physical and Failure Analysis of Integrated Circuits, IPFA (2020)
10. IEEE: 754-2019 - IEEE Standard for Floating-Point Arithmetic. Tech. rep. (2019)
11. Intel Corporation: bfloat16 - Hardware Numerics Definition. Tech. rep. (2018)
12. Kesten, R., Usman, M., Houston, J., Pandya, T., Nadhamuni, K., Ferreira, A., Yuan, M., Low, B., Jain, A., Ondruska, P., Omari, S., Shah, S., Kulkarni, A., Kazakova, A., Tao, C., Platinsky, L., Jiang, W., Shet, V.: Level 5 perception dataset 2020 (2019)
13. Koopman, P., Osyk, B.: Safety argument considerations for public road testing of autonomous vehicles. SAE Technical Papers (April) (2019)
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems (2012)
15. Kuhn, H.W.: The Hungarian method for the assignment problem. Naval Research Logistics Quarterly (1955)
16. Li, G., Hari, S.K.S., Sullivan, M., Tsai, T., Pattabiraman, K., Emer, J., Keckler, S.W.: Understanding error propagation in Deep Learning Neural Network (DNN) accelerators and applications. In: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2017. Association for Computing Machinery, Inc (2017)
17. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollar, P.: Focal Loss for Dense Object Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence (2020)
18. Lotfi, A., Hukerikar, S., Balasubramanian, K., Racunas, P., Saxena, N., Bramley, R., Huang, Y.: Resiliency of automotive object detection networks on GPU architectures. Proceedings - International Test Conference (2019)
19. Microsoft: Coco 2017 dataset (2017)
20. Neale, A., Sachdev, M.: Neutron Radiation Induced Soft Error Rates for an Adjacent-ECC Protected SRAM in 28 nm CMOS. IEEE Transactions on Nuclear Science (2016)
21. Reagen, B., Gupta, U., Pentecost, L., Whatmough, P., Lee, S.K., Mulholland, N., Brooks, D., Wei, G.Y.: Ares: A framework for quantifying the resilience of deep neural networks. Proceedings - Design Automation Conference (2018)
22. Redmon, J., Farhadi, A.: YOLOv3: An Incremental Improvement (2018)

23. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence (2017)
24. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings (2015)