Machine Learning in Space: Surveying the Robustness of on-board ML models to Radiation

Kevin Lange[¶], Federico Fontana[*], Francesco Rossi[*], Mattia Varile[*], Giovanni Apruzzese[¶]
[¶]*University of Liechtenstein*, [*]*AIKO S.r.l.*
{kevin.lange, giovanni.apruzzese}@uni.li[¶], {federico, francesco, mattia}@aikospace.com[*]

***Abstract*—Modern spacecraft are increasingly relying on machine learning (ML). However, physical equipment in space is subject to various natural hazards, such as radiation, which may inhibit the correct operation of computing devices. Despite plenty of evidence showing the damage that naturally-induced faults can cause to ML-related hardware, we observe that the effects of radiation on ML models for space applications are not well-studied. This is a problem: without understanding how ML models are affected by these natural phenomena, it is uncertain "where to start from" to develop radiation-tolerant ML software.**

As ML researchers, we attempt to tackle this dilemma. By partnering up with space-industry practitioners specialized in ML, we perform a reflective analysis of the state of the art. We provide factual evidence that prior work did not thoroughly examine the impact of natural hazards on ML models meant for spacecraft. Then, through a "negative result," we show that some existing open-source technologies can hardly be used by researchers to study the effects of radiation for some applications of ML in satellites. As a constructive step forward, we perform simple experiments showcasing how to leverage current frameworks to assess the robustness of practical ML models for cloud detection against radiation-induced faults. Our evaluation reveals that not all faults are as devastating as claimed by some prior work. By publicly releasing our resources, we provide a foothold—usable by researchers *without access* to spacecraft—for spearheading development of space-tolerant ML models.

I. Introduction

During the last years, machine learning (ML) solutions for on-board satellite missions have seen a tremendous push from academia, space agencies, as well as industry [1]. Indeed, ML can now be used to carry out many space-related tasks (Fig. 1). For instance, thanks to ML, companies in this market can optimize downlink communications, thereby reducing the amount of unusable data and improving efficiency [2]. In addition, the increasing interest in space has reduced the time required to launch satellites in orbit—especially for CubeSats [3], which represent the state of the art of modern spacecraft [4, 5].

CubeSats are typically equipped with commercial-off-the-shelf (COTS) components (e.g., the NVIDIA Jetson Nano), which can be used to empower ML models [6]. Aside from being cheaper [5], some COTS components can also outperform (e.g., faster processing speed) specialized space-tolerant counterparts [7]. Unfortunately, COTS components have a shorter expected lifetime since they are not designed to withstand the harsh space environment [8]. For instance, compared to radiation hardened components (like the NanoeXplore BRAVE Large or Xilinx Kintex XQRKU060), the Myriad2 can withstand only half of the total ionizing dose [9]. Furthermore, Slater et al. found that the NVIDIA Jetson Nano is expected to last at most two years in low Earth orbit [10]; whereas Rodriguez et al. [11] also found that the NVIDIA Xaview SoC is susceptible to faults induced by natural hazards in space.

These hazards, such as radiation, are likely to endanger next-generation COTS components even more, due to the manufacturers' interest in *hardware miniaturization*—which exacerbates the vulnerabilities of COTS components to faults such as bit-flips [11]. Worryingly, abundant prior work (e.g., [12–14]) showed the disruptive impact that bit-flips can have against some ML models. Therefore, there is a need for **fault-tolerant software**, which calls for contributions from *various research domains* (e.g., space, but also applied ML) [15].

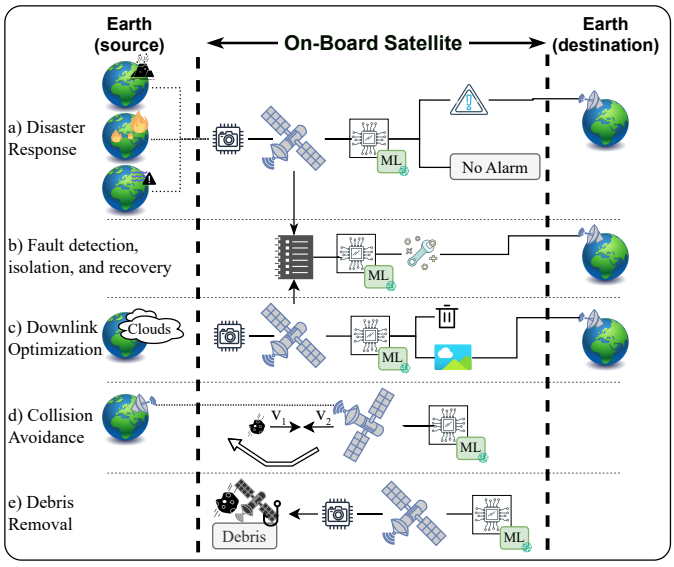


Fig. 1: **Applications of ML on-board spacecraft** – In some cases, ML is used to analyse data *of* Earth (taken by the satellite), and then send the results *to* Earth; in other cases, the output of ML is used by the satellite itself.

In this context, we find ourselves in a quandary. On the one hand, a large body of literature (e.g., [13]) demonstrated the impact of hardware-related faults on ML models. On the other hand, many papers showed that natural hazards in space can lead to damaged hardware (e.g., [10]). However, we ask ourselves: "what about papers that specifically focus on the impact of *natural hazards on ML models meant for on-board deployment in spacecraft*?" Indeed, COTS components can empower many computational elements besides ML models; whereas not all the faults that can affect an ML model at the hardware level may be related to natural hazards in

space. Hence, before developing fault tolerant software for ML components in space, it is first necessary to determine how much the space environment can affect ML models deployed on-board satellites. We pursue this quest in this paper.

CONTRIBUTIONS. We aim to foster development ML methods that are robust to "natural" space hazards. After summarising the applications and problems that entail deployment of ML on satellites (§II), we make the following contributions.

- By performing a thorough *literature review*, we show that **prior work poorly accounted for the effects of natural faults** on ML applications in space (§III); [major contribution]
- To address this issue, we *analyse open-source toolkits* for carrying out simulations of the effects that radiation can have on ML: through *negative experiments*, we find that **some current solutions have functional issues** (§IV).
- To fix this problem, we carry out some **technical experiments** – under the guidance of practitioners – showcasing how to *approximate realistic evaluations*, and the *impact of radiation-induced faults* on ML methods (§V).

Lastly, after outlining implications for related work (§VI), we *publicly release our resources* [16]. Besides doing so for scientific reproducibility and transparency, our tools serve to kickstart future experimentation on space-tolerant ML models—without the need to carry out field tests.

Remark: our contributions should *not be taken as a finger-pointing attempt*. Rather, we perform a reflective exercise on the current landscape of papers and technologies for on-board deployment of ML, with the ultimate objective of improving the state of the art.

II. APPLICATIONS (AND PROBLEMS) OF ML IN SPACE

To setup the stage for our contribution, we summarize the pros-and-cons that entail deployment of ML on-board satellites.

Goal and Audience. We seek to build a bridge between two communities: *space researchers*, who have expertise in studying (often via real equipment) the natural phenomena affecting spacecraft; and *ML researchers*, whose proficiency lies in analysing (typically with open-source resources) the ins-and-outs of ML (see Fig. 2).

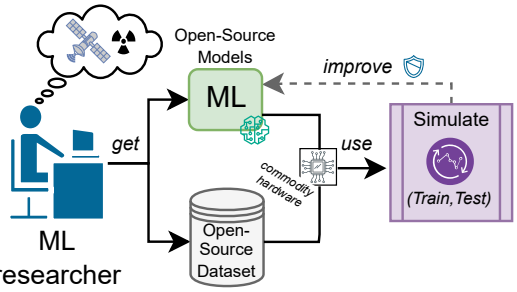


Fig. 2: **Perspective of the ML researcher** – ML researchers do not have access to spacecraft or to physical equipment that reproduces a space setting. They only rely on open-source tools (models and data) and commodity hardware (e.g., GPUs), but their knowhow can help improve state-of-the-art methods for real-world deployments of ML.

A. What tasks can be solved by ML (in space)?

Applications of ML for on-board inference in spacecraft involve the processing of information that can be either *used by the satellite itself* or *sent back to Earth*. Indeed, it has been found [17] that elaborating the data directly on spacecraft provides many benefits. We identify five exemplary applications (refer to Fig. 1) of ML for on-board deployment in Earth-orbit satellites—taken from both research and practice.

a) *Disaster response*. In these contexts, latency is paramount and can be reduced by transmitting only the most essential information [18]. For instance, Ruzicka et al. [19] used ML to identify the areas affected by earthquakes, fires, floods and other natural disasters.

b) *Fault detection, isolation and recovery*. In these cases, the spacecraft uses ML to, e.g., identify faults and issues in its internal pipelines and apply mitigation [7, 20].

c) *Downlink optimization*. One pioneering example is the Φ-Sat-1 Mission from the European Space Agency in 2020 [21]. Here, ML was used to detect clouds in images [22], so as to avoid sending useless "noisy" images (showing mostly clouds) back to Earth, saving bandwidth.

d) *Collision avoidance*. Due to the growing number of satellites in Earth orbit and the increasing risk of potential collisions between satellites, avoiding such situations becomes increasingly relevant. Gonzalo et al. [23] as well as Bourriez et al. [24] showed how to address this problem with ML for on-board computation.

e) *Debris removal*. A fundamental part of mission planning is identifying which targets must be removed, and then pinpoint the most efficient removal order. Notable efforts are the solution by Xu et al. [25] and that by Guthriel [26] (which relies on convolutional neural networks).

We also mention some orthogonal applications of ML, e.g., "federated learning" approaches [27]; or for missions that go beyond the Earth orbit, such as using ML to pinpoint landing on the Moon [28], or for managing the autonomy of CubeSats in deep space [29]. These works are outside our scope.

B. What space-specific problems affect ML (in space)?

Computing hardware deployed on-board a satellite is exposed to a harsh environment. In particular, two "natural hazards" – which are much more present in space than on Earth – can interfere with on-board equipment: temperature and radiation (see Fig. 3 for a schema of these hazards for cloud detection).

*1) **Temperature**:* In-orbit satellites are exposed to extreme temperatures—both cold and hot. In particular, the surface temperature of low-orbit satellites can vary between $[-150; +150]$ Celsius degrees [30]. Heat sources include: the Sun, the Earth's albedo, the infrared emissions of Earth [31]; as well as the heat produced by the satellite itself.

*2) **Radiation**:* Within our solar system we can distinguish three different sources of radiation [32]: solar radiation, which comes from the Sun in the form of solar wind, solar flares and coronal mass injections [33]; galactic cosmic rays, coming from outside our solar system, which are remnants particles of galaxies and stars (while most particles are blocked from the heliosphere, others can reach Earth and affect electronic components in satellites [32]); and the "Van Allen Belt", a toroidal-shaped area of charged particles (whose kinetic energy depends on the Sun's activity [32, 34]) which can be found around planets with a magnetic field—such as Earth.

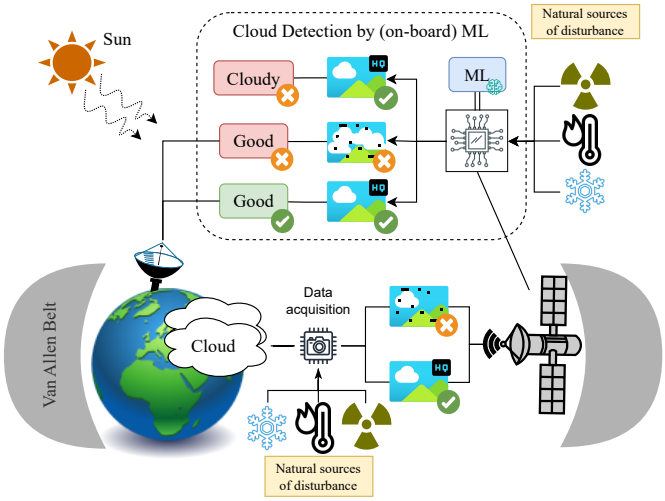


Fig. 3: **Using ML for on-board Cloud Detection** – The satellite acquires data (i.e., images) of Earth; such data may be subject to natural disturbances (e.g., radiation). Then, the captured data is analyzed by an ML model (which may also be subject natural disturbances). The output of this analysis is then sent back to Earth. To optimize downlink communications, "cloudy" images (detected via on-board ML) are not transferred. This saves bandwidth.

The effects of temperature on hardware are well-known in computer science (e.g., [35–37]), so in the next section we focus on radiation—which is intrinsic of space missions.

C. How can radiation interfere with ML (in-space)?

Whenever electronic components are exposed to radiation, ionizing particles continuously interact with the device's semiconductors [38], potentially leading to equipment malfunctions—thereby impacting the ML pipeline in various ways.

*1) **Effects on the ML model**:* While low-energy ions may not have any effect [39], others can cause "single-event transients", manifested through memory bit-flips—which may persist until the memory is overwritten (e.g., for the NASA mission Orbview-2, a state recorder had over 200 daily bit-flips [40]). Although bit-flips not necessarily lead to negative consequences, radiation can interfere with an operation causing a "single event functional interrupt", which can lead to an incorrect output or a system crash [41]. It is even possible for a component to be permanently damaged (i.e., "single-event latchups" [42]), e.g., due to high currents which overheat the circuits [32]. Altogether, these effects can inhibit the correct operation of the device empowering the ML model—either via hardware- or software-faults [12, 13]. The problem is aggravated for COTS equipment (and its miniaturization): the smaller the manufacturing technology used, the lower the amount of radiation such a technology can tolerate [9].

*2) **Effects on Data**:* Radiation can also affect the data that is meant to be further processed by an ML model. For instance, radiation can lower the picture quality [43] of image sensors reliant on CMOS (which are increasingly used for space applications [44]), thereby impacting the accuracy of a cascading ML model. Moreover, sensors' prolonged exposure to radiation increases dark-current in the acquired images [45], potentially leading to permanent damages affecting all taken images [46, 47]. Finally, since radiation can cause overheating [32], it can also lead to the complete loss of the camera (such an issue likely affected the Juno mission [48]). We have created a set of images (see Fig. 4) showcasing the effects of some disturbances applied to images taken by satellites.

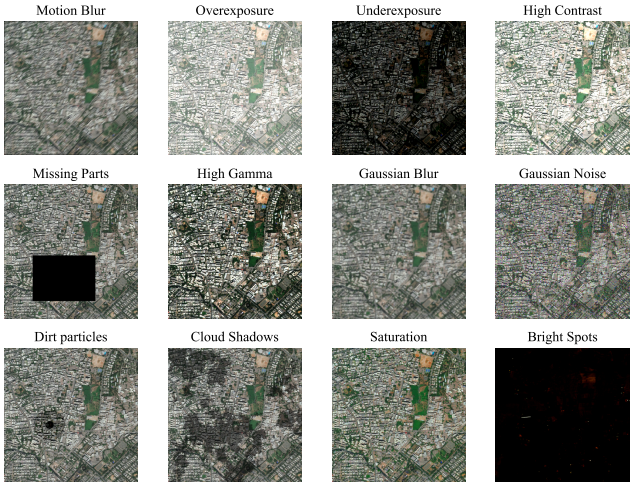


Fig. 4: **Possible image disturbances** – The data (i.e., images) acquired by in-orbit satellites can be perturbed in many ways. Feeding such data to an ML model may "naturally" impact its performance. (Own figure, code is at: [16])

In summary, there is plenty of evidence showcasing the negative effects that such natural hazards can have on components related to ML in space. Thus, we ask ourselves:

Research Question #1: "how well does prior research on ML applications in space account for its natural hazards?"

III. State of the Art (in research)

We highlighted (§II) that abundant prior work showed: *(i)* that radiation (and excessive temperature) has negative effects on ML-related components; and *(ii)* that such negative effects can impact the performance of ML models. Here, as our first contribution, we scrutinize the extent to which prior research considered the effects of such natural hazards when proposing a ML solutions designed for on-board satellite deployment.[1]

A. Methodology (literature review)

We carry out a systematic literature review. This entire procedure was done by two authors in Dec. 2023; to ensure consistency, we repeated the same procedure in Feb. 2024.

Paper collection. We mostly rely on the snowball method [49]. We began our analysis with the papers by Giuffrida et al. [22] and Bruhn et al. [50], due to their popularity[2]. Next, we provided each of these work as input to ConnectedPapers [51]; and also do a forward/backward snowball search (for both [22, 50]: altogether, these operations

[1] **Why is this important?** Among the goals of research is to provide answers to real-world phenomena. However, the research domain does not have always access to real-world equipment—especially for space-related tasks, the costs of such equipment can be prohibitive. Hence, we can expect that most research in this domain carried out their evaluations through simulations. Answering our (first) research question allows one to determine how well the results of prior research approximate those expected from real-world experiments.

[2] As of Feb., 2024, [22] ([50]) has >100 (>55) citations on Google Scholar.

yielded a set of 265 references, which we further expand with the 97 articles listed in OPS-SAT [52].

Screening. We manually filter all (362) these documents with the intent of identifying suitable candidates for a deeper analysis. Specifically, we *excluded*: presentations/abstracts, non-peer–reviewed documents, and papers published before 2014 (outdated). Plus, since our focus is on ML, we excluded any paper that did not mention the terms "machine/deep learning", "neural network", or "artificial intelligence" either in the title or in the abstract. Afterwards, we carried out a preliminary inspection of the text, looking for papers that *considered ML applications in space and for on-board satellites* (in Earth orbit—i.e., the ones discussed in §II-A). To this purpose, we excluded any paper that did not mention the term "on-board" in the main body; and papers that envision ML applications beyond Earth orbit (e.g., [53]), or do not carry out original ML evaluations (e.g., reviews [54]). Altogether, these operations led us to a set of 62 papers (published between [2018–2023]), to which we add the work by Haser and Förstner [55] which we found thanks to the cooperation with industry practitioners.

In-depth analysis. We manually analyse each of our identified 63 papers by considering six axes: *(i)* what ML *use-case* is being considered? *(ii)* does the paper carry out an *on-board evaluation* (i.e., with in-orbit spacecraft) and, if not, is specialized hardware involved? *(iii)* how many times are the terms "radiation" and "temperature" *mentioned* in the text? *(iv)* are the effects of "radiation" or "temperature" considered in the evaluation (either real, or simulated)? *(v)* does the paper *propose* methods to improve the "robustness" of ML to natural hazards in space? *(vi)* is the *code* publicly available? Altogether, assessing these axes allow one to provide a comprehensive overview of the state of the art of ML applications in space and w.r.t. their consideration of natural hazards, thereby answering our first RQ. We report the results of our analysis in Table I. Before analyzing Table I, however, we reiterate that *our analysis is not a fingerpointing attempt*.

B. Findings (and interpretations)

From Table I, three findings are apparent. Out of 63 papers:

- Only 3 papers (5%) propose solutions to make ML methods more "robust" against natural space hazards.
- 29 papers (46%) *never* mention "radiation", and 41 (65%) papers *never* mention "temperature"
  - and 61 (97%) papers mention "radiation" or "temperature" less than 6 times.
- Only 7 papers (11%) release their implementation.

Let us now analyze these results at a low-level, focusing on "where" each paper carries out its evaluation.

In-orbit evaluations (dark-gray rows). Seven (11%) papers perform evaluations on in-orbit spacecraft. Their results undeniably represent the real world, as they are obviously affected by natural hazards. For instance, Giuffrida et al. [21], consider radiation-hardened FPGA, and conclude that such components are "robust"; however, such a statement is derived by simply observing that the resulting performance of the ML models is high despite the exposure to radiation. As a matter of fact,

TABLE I: **Papers on ML applications onboard spacecraft** – We used the snowball method starting from [22, 50] and adding [52]: from 362 documents, we distill 63 papers. Use-cases are taken from §II-A (ColAvo=Collision Avoidance, DisRes=Disaster Response, DebRem=Debris Removal, DlkOpt=Downlink Optimization, FDIR=Fault Detection Isolation Recovery); parentheses denote potential misalignment between testbed and use-case. Row color denotes the experimental settings (white=software, light-gray=space-related hardware, dark-gray=in-orbit spacecraft). In "Rad?" and "Temp?", we report the number of times "radiation" and "temperature" occur in the text; and a ✓ (✗) denotes whether they affect the results (or not). For "robust", the paper had to *evaluate* some hardening/robustness method (not just "mention") to get a ✓. For papers that release "Code", the ✓ points to the public repository; 🗎 is for data only.

Paper (1st author)	Year	Use-Case	Rad?	Temp?	Robust?	Code?
Li [56]	2018	DlkOpt	5 (✗)	5 (✗)		
Pastena [57]	2019	DlkOpt	0 (✗)	0 (✗)		
Esposito [58]	2019	DisRes	0 (✗)	4 (✗)		
Lemaire [59]	2020	DlkOpt	0 (✗)	0 (✗)		
Furano [60]	2020	DlkOpt	36 (✓)	4 (✗)		
Kothari [7]	2020	DlkOpt	6 (✓)	1 (✗)		
Denby [61]	2020	DlkOpt	2 (✓)	4 (✓)		✓
Maskey [62]	2020	DlkOpt	0 (✗)	0 (✗)		
Giuffrida [22]	2020	DlkOpt	7 (✓)	0 (✓)		
Bruhn [50]	2020	DlkOpt	34 (✓)	1 (✓)	✓	
Reiter [63]	2020	DlkOpt	7 (✗)	2 (✗)		
Mateo-Gracìa [64]	2021	DisRes	0 (✗)	0 (✗)		✓
Giuffrida [21]	2021	DlkOpt	9 (✓)	1 (✓)		
Rapuano [65]	2021	DlkOpt	13 (✗)	1 (✗)		
Del Rosso [66]	2021	DisRes	0 (✗)	3 (✗)		🗎
Kucik [67]	2021	DisRes	1 (✗)	0 (✗)		✓
Diana [68]	2021	DlkOpt	14 (✗)	0 (✗)		
Ferrari [69]	2021	DlkOpt	0 (✗)	0 (✗)		
Pacini [70]	2021	DlkOpt	1 (✗)	0 (✗)		
Leong [71]	2021	DlkOpt	0 (✗)	0 (✗)		
Fernando [72]	2021	DisRes	0 (✗)	0 (✗)		
Garrett [73]	2021	(DebRem)	7 (✓)	0 (✗)	✓	
Haser [55]	2022	(FDIR)	10 (✓)	0 (✗)		
Farr [74]	2022	DlkOpt	0 (✗)	0 (✗)		
Růžička [75]	2022	DisRes	0 (✗)	0 (✗)		✓
Azami [76]	2022	DisRes	10 (✓)	12 (✗)		
Luo [77]	2022	DlkOpt	0 (✗)	2 (✗)		
Labrèche [78]	2022	DlkOpt/FDIR	2 (✓)	0 (✓)		✓
Spiller [79]	2022	DisRes	1 (✗)	2 (✗)		
Luo [80]	2022	DlkOpt	0 (✗)	1 (✗)		
Salazar [81]	2022	DlkOpt	0 (✗)	0 (✗)		
Pitonak [82]	2022	DlkOpt	3 (✗)	1 (✗)		
Guerrisi [83]	2022	DlkOpt	0 (✗)	0 (✗)		
Jeon [84]	2022	DlkOpt	0 (✗)	0 (✗)		
Zeleke [85]	2022	DlkOpt	1 (✗)	0 (✗)		
Del Rosso [86]	2022	DisRes	0 (✗)	1 (✗)		🗎
Murphy [87]	2022	FDIR	1 (✗)	2 (✗)		
Spiller [88]	2022	DisRes	0 (✗)	4 (✗)		
Buckley [89]	2022	DlkOpt	31 (✓)	8 (✓)		
Mateo-Gracìa [90]	2023	DisRes	1 (✓)	0 (✓)		✓
Labrèche [91]	2022	DlkOpt	0 (✗)	0 (✗)		✓
Abderrahmane [136]	2022	DlkOpt	0 (✓)	0 (✓)		
Kacker [92]	2022	DlkOpt	3 (✓)	7 (✓)		
Mladenov [93]	2022	DlkOpt	1 (✗)	0 (✗)		
Kervennic [94]	2022	DlkOpt	0 (✗)	0 (✗)		
Fratini [95]	2022	DlkOpt	0 (✗)	0 (✗)		
Gu [96]	2023	DlkOpt	0 (✗)	0 (✗)		
Guerrisi [97]	2023	DlkOpt	1 (✗)	0 (✗)		
Cascelli [98]	2023	DlkOpt	1 (✗)	0 (✗)		
Coca [99]	2023	DisRes	4 (✗)	1 (✗)		
Shi [100]	2023	DlkOpt	1 (✓)	0 (✗)	✓	
Serief [101]	2023	DlkOpt	2 (✗)	0 (✗)		
Kadway [102]	2023	DlkOpt	0 (✗)	0 (✗)		
Ferrante [103]	2023	FDIR	1 (✗)	16 (✗)		
Carbone [104]	2023	DlkOpt	1 (✗)	0 (✗)		
Ciardi [105]	2023	FDIR	7 (✗)	0 (✗)		
Deticio [106]	2023	DlkOpt	0 (✗)	0 (✗)		
Deticio [107]	2023	DlkOpt	0 (✗)	0 (✗)		
Leon [108]	2023	ColAvo	3 (✗)	0 (✗)		
Fernando [109]	2023	DlkOpt	0 (✗)	0 (✗)		
Murphy [110]	2023	FDIR	0 (✗)	0 (✗)		
Nalepa [111]	2023	FDIR	0 (✗)	0 (✗)		
Bourriez [24]	2023	ColAvo	1 (✗)	0 (✗)		

these works *do not underscore the impact of natural hazards* on the corresponding results (i.e., there is no "hazard-free" baseline that can be used to study the effects of such hazards). Indeed, these papers *mention* the term "radiation" (typically to account for limitations of the results which are likely affected by radiation), but do not study this phenomenon.

Hardware-reliant simulations (light-gray rows). The majority (30, 47%) of papers employ space-related hardware in their assessments. Among these, the most "accessible" work to ML researchers is the one by Bruhn et al. [50] which focuses on GPU testing, but the datasets considered (i.e., `MNIST`) have little relevance with space settings. Some simulations carried out on space-compliant hardware typically envisioned in space settings do not consider radiation-tolerant COTS components. For instance, Rosso et al. [66] experiment on the Myriad2, which is weak to radiation (as shown in [9]); the opposite is done, e.g., in Pitonak et al. [82], whose experiments are run on radiation-hardened FPGA—but in both cases, there is no assessment of the actual effects of radiation. Notably, Azami et al. [76] carry out a radiation test for six hours on a Raspberry Pi, showing that a single event latchup occurred after 5hours which increased the power consumption—but there was no measure of how it affected the performance of the ML model. A similar (and far more realistic) experiment was done by Buckley et al. [89]—but even here, the focus was more on the response of the Myriad2 rather than on the performance of the ML model. Nevertheless, *such tests cannot be replicated by a ML researcher* without specialized equipment (and the code of [50, 76, 89] is not public). Notably, however, two papers[3] by Del Rosso et al [66, 86] release their data.

Software-based experiments (white rows). A total of 26 (41%) papers carry out evaluations at the software-level (requiring only, e.g., a GPU). However, most of such papers simply aim at improving the baseline accuracy of ML techniques *and do not account for radiation*. Remarkably, Garrett and George [73] seek to improve the robustness of tensorflow-based ML models to radiation. However, despite mentioning various ML applications (e.g., "debris tracking"), the experiments were carried out on common benchmark datasets (i.e., `MNIST`), which are not representative of a realistic space environment—a limitation affecting also [55]. Furthermore, the code of [55, 73, 100] is also not available—which is, unfortunately, a common occurrence in our analysis.[4]

C. Consequences (and the way forward)

Our analysis reveals a fundamental problem: not only *(i)* most prior research *does not account* for the effects of natural hazards on ML applications in space; but also *(ii)* those papers for which the effects are implicitly considered (e.g., [22, 50]) *do not allow one* to determine their impact on the ML's output; and *(iii) few works openly release* their implementation. Put differently, from a research perspective, it is currently unknown how to approximate (and, hence, reproduce) the effects that natural hazards have on real-world deployments of ML for on-board satellites. This is the source of our quandary.

TAKEAWAYS. Only few works considered the effects that space-natural hazards can have on ML models,[a] and few papers share their resources—hindering reproducibility.

[a]Even a recent "critical analysis" poorly accounts for radiation [112].

If one can truly measure the effects of such hazards, one can also determine which solution can be used to mitigate their impact. Indeed, there exist many techniques that specifically focus on improving the robustness of ML models against bit-flips (e.g., selective hardening [113]). However, all such techniques have been proposed for tasks that do not strictly pertain to ML applications in space. For instance, [114] focuses on healthcare; whereas [115] considers generic object recognition (not in space); furthermore, the experiments in [113] are carried out on `MNIST`. Hence, we ask ourselves:

Research Question #2: "can an ML researcher – without access to specialized equipment – reproduce the effects of natural hazards, and specifically of *radiation*, on ML applications in space by leveraging current technologies?"

IV. Tools for reproducing Radiation's effects

We now analyse the publicly available toolkits that a researcher can use to replicate a realistic environment. In particular, we consider the task of downlink optimization via *cloud detection*, and we focus on the impact that *radiation* can have on the corresponding ML pipeline either by bit-flips or by image distortion.[5] Importantly, for this analysis **we rely on the know-how of** AIKO S.r.l., an European company specialized in the development (and deployment) of ML in spacecraft.

A. Resources for replicating on-board cloud detection via ML

Assume that we want to carry out realistic experiments on ML applications in space, focusing on cloud detection (see Fig. 3), but without having access to a real spacecraft or physical equipment (e.g., [14, 50, 116]). To achieve such a goal, we must: *(i)* get an ML model—potentially by drawing from state-of-the-art methods; *(ii)* train it over one dataset—potentially by ensuring that it achieves state-of-the-art performance; *(iii)* and then find a way to replicate the effects of radiation by either *(a)* manipulating the input test-data, or by *(b)* introducing bit-flips that affect the processing of the ML model (see Fig. 2).

[3]Such data was obtained with drones capturing images from Earth. We considered these as "hardware" and not "real-space" because the altitude of the drone was not high enough to be exposed to the natural hazards of space.

[4]**Disclaimer:** We stress that many works do provide extensive details for reproducibility. For instance, Fratini et al. [95] leverage the NanoSat framework and abundant details are provided in the paper. Hence, lack of source code does not prevent scientific reproducibiilty—despite inevitably delaying further developments (at least from the perspective of ML researchers).

[5]**Why do we focus on this?** The problem of on-board cloud detection via ML is *popular* in prior research (among the 42 papers within the domain of downlink optimization in Table I, 20 consider it); furthermore, we consider radiation-induced bit-flips because we do not want to *physically damage* our own (real) hardware—plus, single events latchups, while possible, are rare in space missions [9] (confirmed by our practitioners); nonetheless, since this task entails analysing images, we can reproduce the effects of (partially) broken sensors on the acquired data. We stress that *our analysis is just a first step*, and we do not claim generality (albeit our findings are applicable also to other use-cases, e.g., disaster response).

Space-specific resources. For the first two steps, we can certainly rely on well-known tools within the ML community: e.g., TensorFlow/PyTorch frameworks, or the `ImageNet` dataset.[6] However, we highlight a repository [117] (which has over 7k stars as of January 2024) that provides plenty of resources for space-related experiments (such as the `95-Cloud` dataset [118]). Unfortunately, after extensively analysing this repository we found that *it provides no suggestion/tool to replicate the effects of radiation* (and not even of extreme temperatures) on ML components. Even recently proposed space-specific "testing labs/implementations" do not envision either ML (e.g. [119]) or the effects of natural hazards (e.g., [120]). We even inquired practitioners if they were aware of any open-source and space-specific platform that provided such a functionality, and they were not aware of any such tool.

Tools for fault-injection. According to practitioners, the only viable way to replicate the effects of radiation (and of natural hazards in general) is to: *(i)* "assume" that a given component is exposed to non-negligible radiation (see §II-C), *(ii)* "guess" the effects of such radiation on the given component; and then *(iii)* manually "inject" the corresponding fault/disturbance. For this purpose, we found[7] three notable tools that can be leveraged for on-board cloud detection.

- NVIDIA Binary Instrumentation Tool for Fault Injection (`NVBitFI`) [121] (paper: [122]), useful to *emulate* faults at the hardware level (e.g., on the GPU accelerator);
- Low-Level Tensor Fault Injector (`LLTFI`) [123] (paper: [12], useful to *simulate* faults at the software level (e.g., by operating on the ML model during its analysis).
- `Kornia` [124] (paper: [125]), which is a library for image augmentation and hence useful to inject disturbances in images (we used Kornia to make Fig. 4).

We also mention `PyTorchFI` [126] and `TensorFI` [127], which are specifically tailored for PyTorch and TensorFlow. However, after inquiring practitioners, they discouraged from using these tools since they operate at a much higher level than, e.g., `LLFTI`; furthermore, `TensorFI`'s (whose last commit in its repo dates back to 2021) does not support Python3 and TensorFlow2—which are the state of the art for ML.

B. Analysis of `NVBitFI` and `LLTFI` (negative result)

We now seek to use the identified tools to carry out a proof-of-concept evaluation of on-board ML for cloud detection. Ideally, faults injected at the hardware level are more realistic, since cosmic radiation affects physical equipment. Hence, a researcher interested in replicating a real setting for on-board cloud detection via ML would first opt for `NVBitFI`, and only consider `LLTFI` as a last resort.

1) ***`NVBitFI` (scarce documentation):*** After reviewing the repository of `NVBitFI` [121] (and its research paper [122]) we found that, in its documentation, *there are no instructions on how to use* `NVBitFI` for ML-specific experiments. We tried to infer some low-level details from papers that claim to have used `NVBitFI` for ML evaluations, but we could not find any meaningful instruction: most papers (e.g., [73]) do not release the source-code. The only relevant effort is the paper by Dos Santos et al. [128], providing a forked version of the `NVBitFI` repository which includes some "test-apps" that also contain ML models: however, these ML models are not functional (they are either old, or written in C) for space-related applications, and there is scarce documentation on how to setup the environment to develop alternative ML models. We hence posit that, although `NVBitFI` can theoretically be used for ML experiments, its current implementation can hardly support space-specific evaluations. Even by inquiring practitioners, we were told that they did not know how to setup a space-compliant and ML-ready testbed through `NVBitFI`.

2) ***`LLTFI` (problems and errors):*** We then turn our attention to `LLTFI`, which is specifically designed for ML experimentation. The documentation in the `LLTFI` repository [123] provides detailed instructions, which we rigorously follow. Specifically, after installing the `LLTFI` framework, we use TensorFlow to train an U-Net (an image-segmentation ML model, representing the state-of-the-art for cloud detection [129]) on the `95-Cloud` dataset [118] (we apply an 80:20 train-test split). After having trained the model (and verifying that its performance aligns with the state of the art on this dataset), we follow the guidelines of `LLTFI` and successfully convert our model in ONNX format. However, from now on we began to encounter "unexpected" issues (see Fig. 5).

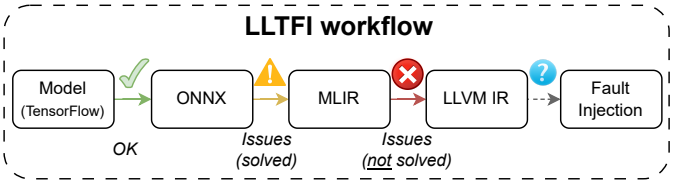


Fig. 5: **Our negative experiment** – We followed the guidelines provided by the developers of `LLTFI`. We could not finish the workflow due to a fatal error for which we found no workaround (even after consultation with practitioners).

- *ONNX→MLIR*: before injecting the faults, `LLTFI` requires the model (in ONNX format) to be converted into its MLIR representation, which can be done with an ONNX-MLIR conversion-tool linked in the repository [130]. We were not able to complete this procedure due to an error ("unhandled option in ConvTranspose"), which we hypothesized was related to the Conv2DTranspose layer. We tried changing this layer to an UpSampling2D (envisioned in the U-Net used, e.g., in [131]), and we also were not successful ("not implemented yet", suggesting that this layer was not supported). We found a workaround by using the most-recent version of the ONNX-MLIR tool [132]—*not provided in the* `LLTFI` repository.
- *MLIR→LLVM-IR*: `LLTFI` injects the faults at a low level, which requires the MLIR version of the model to be further converted into LLVM-IR. We were not able to complete this procedure with the instructions in `LLTFI`'s repository: when we run the commands in the docs, we encounter an error ("operation being parsed with an unregistered dialect"),

[6] One can also use the code developed by prior research. Unfortunately, as shown in Table I, few papers publicly release their artifacts.

[7] We perform this search by looking at the frameworks mentioned in prior work, as well as by manually searching over the repositories linked in [117].

and upon following the provided suggestion we encounter another error ("custom op 'memref.dim' is unknown") which we could not troubleshoot.

We searched in the discussion section of the `LLTFI` repository, but we could not find a solution. After consultation with practitioners,[8] even they were not able to resolve the issue. This "failed experiment" is provided in our repository [16].

C. Reflections and remediations

Our analysis leads us to derive two takeaways—but we reiterate that we are *not pointing-the-finger* (see disclaimer §I):

> **TAKEAWAYS.** (1) There is a lack of open-source resources that facilitate realistic experiments for on-board ML. (2) The few existing toolkits do not allow to setup a testbed that supports state-of-the-art ML methods for cloud detection.

Inspired by these (negative) findings – which lead to an unsatisfactory answer to our RQ#2 – we posit that *it is still possible* to assess the effects of (radiation-induced) faults on a representative ML pipeline for cloud detection. This requires to introduce such "bugs" by manually tampering with the (trained) ML model—i.e., by directly manipulating its weights (thereby simulating a bit-flip [13]). We hence ask ourselves:

> **Research Question #3:** "What are some possible effects that *(i)* manual manipulation of the *ML model's weights*, as well as *(ii)* various *disturbances of the input* images – both of which hypothetically resembling radiation-induced faults – have on the performance of the ML model?"

V. Technical Implementation and Assessment

As a constructive step-forward, our third contribution entails showcasing the effects of exemplary radiation-induced faults through original experiments—and releasing our developed source-code [16]. To this purpose, we first develop our baseline ML models for cloud detection (§V-A); then, we examine their robustness to manually introduced bit-flips (§V-B); finally, we scrutinize their response when the faults affect the input test-data (§V-C). *Our evaluation is a proof-of-concept.*[9] We carry out our experiments on a system mounting an Intel-Core i9 12900K with 32GB of RAM and an NVidia RTX3060Ti.

A. Baseline: U-Net on Cloud-95 dataset

We align our assessment with our "failed" experiment (§IV-B).

Setup. We rely on the U-Net (shown in Fig. 6): according to practitioners, this is the ML model they deploy on spacecraft for cloud detection. The dataset of choice is `95-Clouds`, from which we derive a train, $\mathbb{T}$, and test, $\mathbb{E}$, partition by applying an 80:20 split (common in related work [133]), which correspond to roughly 16k and 4k samples (after filtering-out some "incorrect" samples). Then, we extract a subset, $\mathbb{R}$, of 50 samples from $\mathbb{E}$: we will use this subset as a basis for our robustness assessment. This is necessary for a broad and comprehensive evaluation: in the real world, the effects of natural hazards are not deterministic (e.g., it is impossible to predict how any given sample may be affected by, say, some incorrect pixels). By considering such a small subset, we can gauge the impact of various types of faults (performing our assessment on the whole $\mathbb{E}$ would have required 80x longer runtime—amounting to several weeks of no-stop computing). Finally, to mitigate experimental bias, *we develop three* U-Nets by keeping the same $\mathbb{T}$, $\mathbb{E}$, $\mathbb{R}$, but by repeating the training from scratch with different randomly-initialized weights (repeating trials is fundamental in ML research). We set our the learning phase of our U-Nets to last for ≈3 hours, after which we stop (to avoid overfitting) and proceed with the assessment.

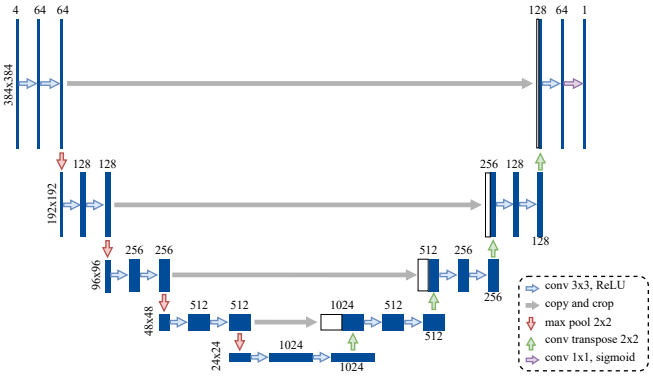


Fig. 6: **Architecture of our U-Nets** – The ML models used for our original experiments are drawn from the state of the art for cloud detection [129].

Results. We report the baseline performance of our ML models (after learning) in Table II, showing the accuracy, precision and recall of our U-Nets on $\mathbb{T}$, $\mathbb{E}$ and $\mathbb{R}$. We appreciate that, on average, our ML models achieve good performance on the test set (i.e., $\mathbb{E}$), and comparable with state-of-the-art works (e.g., [131]). These findings confirm *our baselines are a solid choice for our robustness assessment*. Our repository includes the detailed implementation [16].

TABLE II: **Baseline –** We train three U-Nets (on $\mathbb{T}$ for 3h). Then, we compute accuracy (*Acc*), precision (*Pre*) and recall (*Rec*) on our selected subsets ($\mathbb{T}$, $\mathbb{E}$, $\mathbb{R}$) of `95-Cloud`. Our results (on $\mathbb{E}$) match those of prior works (e.g., [131]).

Model	Training Set ($\mathbb{T}$)			Test Set ($\mathbb{E}$)			Robustness Set ($\mathbb{R}$)		
	Acc	*Pre*	*Rec*	*Acc*	*Pre*	*Rec*	*Acc*	*Pre*	*Rec*
U-Net #1	90.1	92.9	82.9	89.9	92.5	81.7	91.2	93.5	84.1
U-Net #2	89.1	83.3	92.4	88.9	82.8	91.6	91.6	87.7	93.0
U-Net #3	88.1	81.5	92.7	87.8	81.0	92.0	90.4	84.7	94.8
average	89.1	85.9	89.3	88.9	85.4	88.4	91.1	88.6	90.7

B. Simulating radiation-induced bit-flips on the ML models

An elusive property of naturally-occurring bit-flips is the impossibility of predicting *which* and *how many* bits will be

[8] **Practitioners' Feedback:** we inquired the practitioners opinion on (potentially) using `LLTFI` and `NVBitFI` for their simulations meant to improve the robustness of "radiation-hardened" ML prototypes. Accordingly, `NVBitFI` is not ideal, since its workflow poses a high overhead. Indeed, companies typically convert ML models in ONNX for their applications, hence `LLTFI` represents a more suitable solution—if it worked!

[9] **Problem:** we try to "anticipate" what effects radiation may have on the ML pipeline. *Our anticipations are hypothetical*: there is no guarantee that any given sample be modified in the way we do it; and there is also no guarantee that ML model deployed on-board be impacted by natural radiation in the way we do it. We carry out our evaluation under the guidance of practitioners.

flipped—i.e., each bit has the same probability of being flipped as any other bit. Hence, we consider various bit-flips.[10]

Setup. We take each (trained) U-Net, and we manually modify its learned weights and biases in three ways: first, a worst-case scenario (we provide an example in Fig. 7) wherein we modify the bits of the *exponent* (ExpBF); and two others wherein we modify the bits controlling the *mantissa* (ManBF) or the *sign* (SgnBF). Depending on the weight, ManBF can be more (or less) impactful than SgnBF. For a bias-free assessment, we consider bit-flips of one bit, which are chosen *randomly* among the weights of each layer of our baseline models—each having 44 layers. We then run our "faulty" models again on the samples in the robustness set, $\mathbb{R}$. We repeat this experiment 50 times to account for randomness.[11]

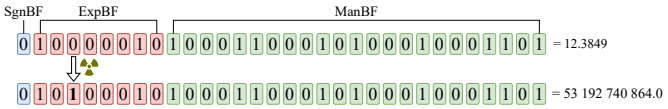
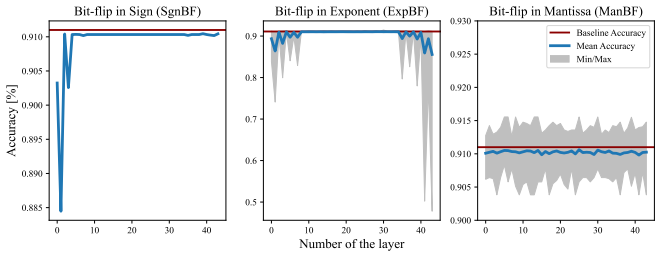


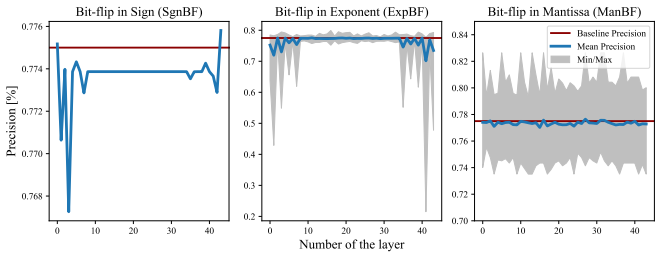
Fig. 7: **Exemplary effects of a bit-flip** – By flipping just a single *exponential* bit (from a 0 to a 1), the value 12.38249 changes to 53 192 740 864.0.

Results. We report the results in Figs. 8, showing the performance (y-axis), averaged across our 3 models, for each type of bit-flip which affects a given layer of our U-Net (x-axis); the lines represent the mean (blue), baseline (red) and the min/max range achieved during the 50 trials. Specifically, Fig 8a focuses on accuracy, Fig. 8b on precision and Fig. 8c on recall. Our repository includes instructions to perform additional assessments to further mitigate bias. By observing the accuracy (which is the most common metric) in Fig. 8a, we see that the majority of our single bit-flips have a negligible impact. Notably, however, for ExpBF the performance of our "faulty" models decreases the most when the bit-flip affects the outer layers of the U-Net; whereas only the first layers are affected in SgnBF—albeit at a much lower degree than for ExpBF. Intriguingly, ManBF may increase the performance.

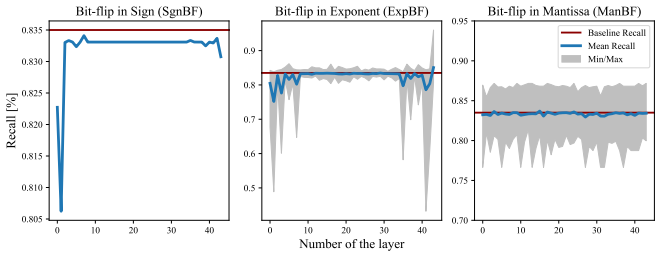
Analysis. We find it instructive to compare this experiment with the one by Haser and Förstner [55]. Specifically, the evaluation in [55] seeks to assess the robustness of neural networks against bit-flips on the `MNIST` dataset (we posit that this setting is highly unrepresentative of a realistic space environment). Nevertheless, the results in [55] show that the accuracy of the ML model drops to that of a coin-toss by flipping 700 bits; and that, e.g., SgnBF affecting over 50% of the layers also leads to an unusable model. According to our practitioners, these findings, while useful, represent "unlikely" circumstances: even in space missions, such high corruptions are unlikely. This is why we opted for single bit-flips, which are more likely to occur and also more subtle. Moreover, Haser and Förstner [55] conclude that "if a corruption occurs in later parts its impact is more harmful then in early layers" (sic). This is in contrast with our findings: e.g., for ExpBF, bit-flips affecting last *and* first layers are more impactful than on middle layers (result confirmed with a statistical t-test: $p \approx 0$).



(a) Accuracy.



(b) Precision.



(c) Recall.

Fig. 8: **Effects of bit-flips** – We measure the performance (averaged across 50 trials) of our bit-flips on each possible layer (out of 44) of our 3 U-Nets.

C. *Effects of radiation-induced perturbations on image data*

Images can be affected by natural hazards in many ways (see Fig. 4 in §II-C2). Here, we only consider those disturbances that are more common in space—according to practitioners.

Setup. We consider three types of disturbances: *hot* pixels, which involve having some pixels of an image be "brighter" w.r.t. their originals; *dark currents*, which yield a fixed-pattern noise in an image; and *radiation streak*, which involve having multiple pixels in succession to (incorrectly) have the same value. According to practitioners, these disturbances are very common in the images captured by operational spacecraft. We provide an exemplary illustration of wherein we combine all these disturbances (alongside their effects on the ML model) in Fig. 9. For each of the 50 samples in $\mathbb{R}$, we apply each disturbance at various noise levels (e.g., we progressively increase the number of faulty pixels in an image); the areas affected are chosen randomly. Our repository provides the detailed implementation [16]. After applying these manipulations, we run our (baseline) U-Nets again on the perturbed images.

Results. We report the results in Fig. 10, showing the average accuracy, precision and recall (y-axis) achieved by our

[10] **Background on bit-flips.** In simple terms, injecting bit-flips in an ML model means taking the bits representing the weights of the ML model and flipping them from a 0 to a 1, or from a 1 to a 0. In theory, changes to the "first" bits are likely to induce a greater impact to the final output of the ML model; whereas changes in the "last" bits are more likely to have a negligible effect. We also stress that a single flip in a "high" bit can be more impactful than multiple changes in "lower" bits. For more detailed information, see: [13].

[11] Overall, we perform 3(models)×44(layers)×3(types)×50(repeats)=19 800 bit-flips. This entire assessment required 10h of non-stop computation.

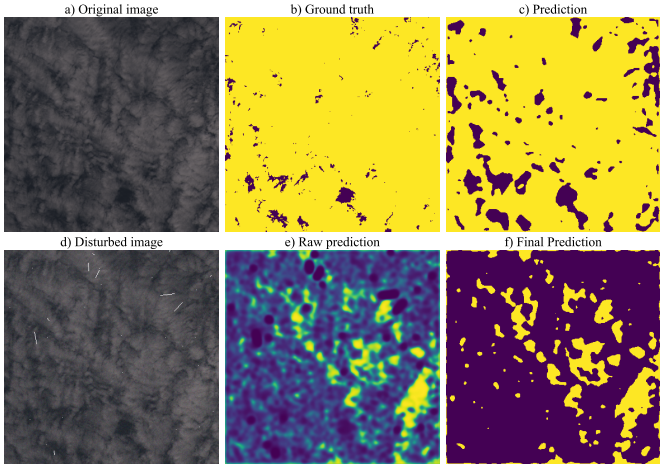


Fig. 9: **Exemplary effect of disturbances** – We show how the effects of introducing common disturbances in spacecrafts (hot pixels, radiation streaks, dark currents) to the input images during the ML model's inference.

three U-Nets at varying intensity (x-axis) of each disturbance applied to all samples in $\mathbb{R}$. Our repository includes the code to run this evaluation on different images, as well as to repeat these experiments at different noise levels and/or more times. From Fig. 10, we observe that the performance decreases for increasing noise levels, especially *dark currents* and *streaks*.

Analysis. We find it interesting, however, that *hot pixels* have a relatively mild effect. This result contrasts the impact of targeted "one-pixel" attacks that are sometimes considered in related papers on adversarial ML robustness [134]. From a security standpoint, this result underscores that radiation-induced faults must be treated differently than "adversarial examples", whose countermeasures (e.g., adversarial training) induce a decrease in the baseline performance of an ML model [135] (which may not be justified in space-contexts).

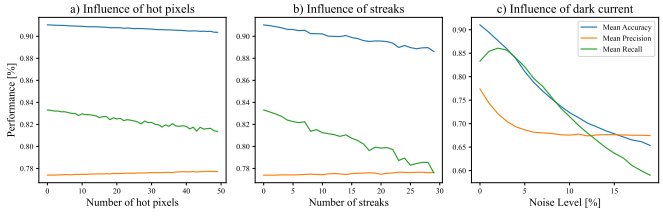


Fig. 10: **Effects of image disturbances** – We compute the performance (averaged across our three U-Net) on the images in $\mathbb{R}$ after applying our disturbances at increasing noise levels.

VI. Discussion and Implications

We now critically review our study, identifying limitations and clarifying how our findings are useful to related research.

A. Disclaimers (and Alternative Formulations)

We acknowledge that our work may have limitations. Let us reflectively scrutinize each of our contributions.

Literature Review (§III). The papers included in Table I have been chosen by two authors who queried popular scientific resources for publications that envisioned applications of ML for on-board spacecraft. For instance, papers proposing an ML method for cloud detection in images but which do not specifically aim (or mention) at on-board satellite deployment, such as [136], were omitted from our analysis. We did so because our goal was assessing the the extent to which prior work on ML for space accounts for its natural hazards. Furthermore, we did not perform the forward/backward search ad infinitum (we inspected 362 documents just from 2 papers), so we did not include papers that matched our inclusion criteria if they were cited (or cited by) in a longer chain. For instance, we did not include the paper by Sabogal et al. [116] (which carries out a detailed assessment on dedicated hardware—including a real radiation-beam!). We also did not consider several works ([137–143]) because they were behind a paywall we could not overcome (even through our own institutions). Nonetheless, our findings revealed that abundant prior work on ML in space poorly accounted for natural hazards, and that few release their resources publicly. However, we acknowledge that there is a growing body of research (notably, from the University of Pittsburgh [144–146]) that is investigating such phenomena. Our work can inspire future endeavours to better account for the issues that affect deployment of ML in spacecraft—and openly release corresponding toolkits to the research community.

Tool Analysis (§IV). Our survey of state-of-the-art technologies for ML-based experiments in space contexts was limited to *publicly available* solutions, which have been found by two researchers supported by practitioners. We are aware that we may have overlooked, e.g., some repositories, and that some closed-source resources (potentially available upon request) may exist which may allow a researcher to simulate realistic testbeds (e.g., [147]). Furthermore, in our analysis we excluded PyTorchFI (used, e.g., in [100]—whose code is not released) because it was deemed impractical by practitioners. Finally, our low-level experiments (for `NVBitFI` and `LLTFI`) have been carried out by individuals who, despite having plenty of experience (over 5 years) in software development and/or ML, may have overlooked some details.[12] Nonetheless, reproducibility issues are—unfortunately—common in the ML domain (e.g., [148]). To fix this, and also for complete transparency, we disclose our code and procedures [16].

Proof-of-Concept Experiments (§V). Our assessment is an effort to spearhead novel research that accounts for radiation-induced faults in space-related evaluations—with the ultimate goal of finding ways to mitigate this real problem. Our results entail the application of well-known methods (U-Net) on benchmark datasets (`95-Cloud`) and by simulating faults (bit-flips or image disturbances). However, we cannot claim our testbed has 100% fidelity with a real spacecraft—albeit the practitioners that we inquired confirmed our workflow to represent a realistic pipeline. Moreover, we do not claim novelty in the methods we use—albeit we do claim our findings to be original (especially in light of the conclusions in [55]). Finally, there are potentially infinite ways to inject

[12] **Ethics:** we contacted the maintainers of the `NVBitFI` and `LLFTI` repositories, informing them of the issues we encountered while using these tools.

the faults we considered. We account for this by releasing our source-code [16]: the interested researcher can replicate our experiments, or carry out new evaluations by considering different faults, models, or datasets.

TAKEAWAYS. Our technical contributions are addressed at ML enthusiasts who want to develop of space-tolerant ML software *without* relying on specialized hardware.[a]

[a] §III and §IV showed that existing solutions have practical limitations.

B. Implications for Research (and for Practice)

We focus on deployment of ML in spacecraft—a setting that partially overlaps with, e.g., aerial vehicles and communications [149–151] or GPS [152]. We focus on solutions at the software-level: developing radiation-tolerant hardware (e.g., [153]) is an orthogonal research field. Our contributions can be leveraged by a wide audience: we discuss 3 groups.

Power Consumption. Many papers on space missions consider the relationship between ML and power consumption [154–156]. For instance, [156] propose to use ML to forecast energy requirements, whereas [154] assess the power consumption of equipment which can also empower ML models. Our contributions can be helpful to this research area: indeed, they would inspire ML researchers "on Earth" to gauge the accuracy applications of ML similar to those envisioned in [156] by simulating certain faults, and then devising appropriate ML-specific hardening methods. Alternatively, since radiation can degrade the performance of ML also *during training* [157], it would be intriguing to assess the energy expenditure of ML models under the impact of radiation-induced bitflips: such an analysis would be useful for those applications that envision on-board ML training [78].

Space Security. We focus on the impact of naturally-induced faults on ML models—a problem that falls in the "robustness" research domain. Our findings are hence closely related to cybersecurity [158]. E.g., papers on "adversarial ML" (e.g., [135]) envisage "data perturbations", which are strongly connected to the perturbations that we injected for our image experiments (§II-C). Notably, some works envision "adversarial attacks" against object detectors meant for space deployment [159], but *do not account for radiation*: it would be intriguing to study the effects of such "adversarial perturbations" if combined to the effects of naturally-occuring perturbations. Finally, satellites are now targeted by real attackers [160], and must hence be protected [161, 162] against sophisticated cyberthreats (e.g., [163, 164]). Our findings (and resources) can open research avenues considering, e.g., the effects of radiation-induced faults on ML models for on-board network security (e.g., [165]).

Developers and Practitioners. Among the main take-home messages of our paper is that there is a lack of publicly available resources for realistic assessments of ML methods in space contexts. Indeed, our work would not have been possible without the guidance of practitioners, who provided us with valuable information on how ML is deployed in spacecraft. Put simply, we argue that *(i)* the shortage of "plug-and-play" (and open-source) tools for radiation-induced faults, paired with *(ii)* the prohibitive costs to setup a representative testbed for space-related assessments is a substantial barrier for research breakthroughs. For instance, the ML community flourished thanks to the open release of code and data; yet, the results claimed by research papers on generic ML have questionable value for deployment of ML on-board satellites, due to the "naturally adversarial" environment of spacecraft—of which we know very little about from the ML perspective. Hence, *we advocate developers at all levels to prioritize publicly accessible and easy-to-use tools for space experiments* (and, in particular, for robustness assessments of ML methods).

Call to Action. To quote Crum et al. [15]: "Progress in this field depends on many stakeholders working together efficiently; not only scientists from the astronomy and physics, ground- and space-based communities, but also engineers, software developers, data, and communication scientists, and more" and "to help development [...] an open simulation and test environment is needed".

VII. Conclusions and Future Work

We seek to improve the robustness of ML models deployed on-board spacecraft to radiation-induced faults. We reveal that *(i)* prior research poorly accounted for the natural hazards that may impact the performance of ML in space; and that *(ii)* current open-source technologies are poorly suited to examine this problem from the perspective of an ML researcher. To improve the current situation, we pair-up with practitioners and carry out proof-of-concept experiments highlighting the effects of some radiation-induced faults on state-of-the-art ML methods for cloud detection. Moreover, to foster development of future efforts focusing on radiation-tolerant ML components, we release all our tools and data in a dedicated repository [16]. We also include a **demonstrative 2-minutes video**, showcasing the simplicity of using our resources.

As intriguing avenues for research that can be built upon this work, we suggest: the assessment of radiation-induced faults *at training-time* (we only considered effects at the inference-stage), since ML models should be periodically updated and re-trained; and the consideration of ML models that analyse data *different from images* (e.g., power or network data).

References

[1] N. Altaf. (2021) The New Space Age. https://www.ibm.com/blog/ibm-develops-a-unique-custom-edge-computing-solution-in-space/. IBM.

[2] D. Bradley and L. Brandon, "Orbital Edge Computing: Machine Inference in Space," *IEEE Computer Architecture Letters*, 2019.

[3] European Space Agency. (2023) CubeSats. https://esa.int/Enabling_Support/Preparing_for_the_Future/Discovery_and_Preparation/CubeSats.

[4] S. W. Samwel, E. A. El-Aziz, H. B. Garrett, A. A. Hady, M. Ibrahim, and M. Y. Amin, "Space radiation impact on smallsats during maximum and minimum solar activity," *Advances in Space Res.*, 2019.

[5] F. Rawlins, R. Baker, and I. Martinovic, "Death By A Thousand COTS: Disrupting Satellite Communications using Low Earth Orbit Constellations," in *SpaceSec (NDSS-W)*, 2023.

[6] M. Lofqvist and J. Cano, "Accelerating Deep Learning Applications in Space," in *Annual Small Satellite Conference (Workshop)*, 2020.

ACKNOWLEDGEMENT. We would like to thank the anonymous reviewers of the SpaceSec workshop for the invaluable feedback. We also thank the Hilti Corporation for funding part of this research.

[7] V. Kothari, E. Liberis, and N. D. Lane, "The final frontier: Deep learning in space," in *Int. Workshop on Mobile Computing Systems and Applications*, 2020.

[8] R. Cantoro, S. Carbonara, A. Floridia, E. Sanchez, M. S. Reorda, and J.-G. Mess, "An analysis of test solutions for COTS-based systems in space applications," in *IEEE Int. Conf. Very Large Scale Integr.*, 2018.

[9] E. Rapuano, G. Meoni, T. Pacini, G. Dinelli, G. Furano, G. Giuffrida, and L. Fanucci, "An FPGA-Based Hardware Accelerator for CNNs Inference on Board Satellites: Benchmarking with Myriad 2-Based Solution for the CloudScout Case Study," *Remote Sensing*, 2021.

[10] W. S. Slater, N. P. Tiwari, T. M. Lovelly, and J. K. Mee, "Total Ionizing Dose Radiation Testing of NVIDIA Jetson Nano GPUs," in *IEEE High Perf. Extreme Comp. Conf.*, 2020.

[11] I. Rodriguez-Ferrandez, M. Tali, L. Kosmidis, M. Rovituso, and D. Steenari, "Sources of Single Event Effects in the NVIDIA Xavier SoC Family under Proton Irradiation," in *IEEE Int. Symp. On-Line Testing and Robust System Design (IOLTS)*, 2022.

[12] U. K. Agarwal, A. Chan, and K. Pattabiraman, "LLTFI: Framework Agnostic Fault Injection for Machine Learning Applications," in *IEEE ISSRE*, 2022.

[13] A. S. Rakin, Z. He, and D. Fan, "Bit-flip attack: Crushing neural network with progressive bit search," in *IEEE/CVF ICCV*, 2019.

[14] E. Miller, C. Heistand, and D. Mishra, "Space-operating linux: An operating system for computer vision on commercial-grade equipment in leo," in *IEEE Aerospace Conference*, 2023.

[15] G. Crum, M. Dosberg, E. Gizzi, C. Gramling, C. Green, J. E. Hill, M. Johnson, K. Mauldin, R. Morgenstern, C. Roberts *et al.*, "Nasa's goddard space flight center's distributed systems missions architecture," in *International Astronautical Congress*, 2022.

[16] https://github.com/langekevin/mlspace_robustness/, 2024.

[17] B. Denby and B. Lucia, "Orbital Edge Computing: Nanosatellite Constellations as a New Class of Computer System," in *ASPLOS*, 2020.

[18] D. Izzo, G. Meoni, P. Gómez, D. Dold, and A. Zoechbauer, "Selected trends in artificial intelligence for space applications," in *Artificial Intelligence for Space (AI4SPACE)*, 2022.

[19] V. Růžička, A. Vaughan, D. De Martini, J. Fulton, V. Salvatelli, C. Bridges, G. Mateo-García, and V. Zantedeschi, "RaVÆn: unsupervised change detection of extreme events using ML on-board satellites," *Scientific Reports*, 2022.

[20] P. Miralles, A. Scannapieco, N. Jagadam, P. Baranwal, B. Faldu, R. Abhang, S. Bhatia, S. Bonnart, I. Bhatnagar, B. Batul, P. Prasad, H. Ortega-González, H. jagan raj, H. More, S. Morchedi, A. Panda, M. Di Fraia, D. Wischert, and D. Stepanova, "Machine Learning in Earth Observation Operations: A review," in *Proc. International Astronautical Congress (IAC)*, 2021.

[21] G. Giuffrida, L. Fanucci, G. Meoni, M. Batič, L. Buckley, A. Dunne, C. van Dijk, M. Esposito, J. Hefele, N. Vercruyssen *et al.*, "The Φ-Sat-1 mission: The first on-board deep neural network demonstrator for satellite earth observation," *IEEE Transactions on Geoscience and Remote Sensing*, 2021.

[22] G. Giuffrida, L. Diana, F. de Gioia, G. Benelli, G. Meoni, M. Donati, and L. Fanucci, "CloudScout: A Deep Neural Network for On-Board Cloud Detection on Hyperspectral Images," *Remote Sensing*, 2020.

[23] J. L. Gonzalo and C. Colombo, "On-board collision avoidance applications based on machine learning and analytical methods," in *Europ. Conf. Space Debris*, 2021.

[24] N. Bourriez, A. Loizeau, and A. F. Abdin, "Spacecraft Autonomous Decision-Planning for Collision Avoidance: a Reinforcement Learning Approach," in *International Astronautical Congress*, 2023.

[25] Y. Xu, X. Liu, R. He, Y. Zhu, Y. Zuo, and L. He, "Active Debris Removal Mission Planning Method Based on Machine Learning," *Mathematics*, 2023.

[26] B. Guthrie, M. Kim, H. Urrutxua, and J. Hare, "Image-based attitude determination of co-orbiting satellites using deep learning technologies," *Aerospace Science and Technology*, 2022.

[27] N. Razmi, B. Matthiesen, A. Dekorsy, and P. Popovski, "On-board federated learning for dense leo constellations," in *International Conference on Communications*, 2022.

[28] S. Silvestrini, M. Piccinin, G. Zanotti, A. Brandonisio, I. Bloise, L. Feruglio, P. Lunghi, M. Lavagna, and M. Varile, "Optical navigation for Lunar landing based on Convolutional Neural Network crater detector," *Aerospace Science and Technology*, 2022.

[29] R. Walker, D. Binns, C. Bramanti, M. Casasco, P. Concari, D. Izzo, D. Feili, P. Fernandez, J. Fernandez, P. Hager, D. Koschny, V. Pesquita, N. Wallace, I. Carnelli, M. Khan, M. Scoubeau, and D. Taubert, "Deep-space CubeSats: thinking inside the box," *Astronomy and Geophysics*, 2018.

[30] P. Gordo, T. Frederico, R. Melicio, S. Duzellier, and A. Amorim, "System for space materials evaluation in LEO environment," *Advances in Space Research*, 2020.

[31] F. Tribak, O. Bendaou, and F. Nejma, "Impact of orbit inclination on heat transfer in a 1U LEO CubeSat," *MATEC Web of Conferences*, vol. 371, 11 2022.

[32] R. Baumann and K. Kruckmeyer, *Radiation Handbook for Electronics*. Texas Instruments, 2019.

[33] K. Tekbıyık, G. K. Kurt, A. R. Ekti, and H. Yanikomeroglu, "Reconfigurable intelligent surfaces in action for nonterrestrial networks," *IEEE Vehicular Technology Magazine*, 2022.

[34] I. Daglis, *Space Storms, Ring Current and Space-Atmosphere Coupling*. Springer Netherlands, 2001.

[35] M. Platini, T. Ropars, B. Pelletier, and N. De Palma, "CPU overheating characterization in HPC systems: a case study," in *IEEE/ACM Workshop Fault Toler. HPC Extr. Scale*, 2018.

[36] M. Sarafraz, A. Arya, F. Hormozi, and V. Nikkhah, "On the convective thermal performance of a CPU cooler working with liquid gallium and CuO/water nanofluid: A comparative study," *Applied Thermal Engineering*, 2017.

[37] H. Handel, "Analyzing the influence of camera temperature on the image acquisition process," in *Three-Dimensional Image Capture and Applications*, 2008.

[38] M. J. Cannizzaro and A. D. George, "Evaluation of RISC-V Silicon Under Neutron Radiation," in *IEEE Aerospace Conference*, 2023.

[39] Y. Liu, G. Armstrong, B. Campanini, S. Messenger, and J. Rodriguez, "Characterization of low dose rate ionizing radiation effect on the micropac 66266-303 optocoupler," in *IEEE Radiation Effects Data Workshop*, 2023.

[40] C. Poivey, J. Barth, K. LaBel, G. Gee, and H. Safren, "In-flight observations of long-term single-event effect (SEE) performance on Orbview-2 solid state recorders (SSR)," in *IEEE Radiation Effects Data Workshop*, 2003.

[41] I. Loskutov, N. Kravchenko, V. Marfin, P. Nekrasov, D. Bobrovsky, A. Smolin, and A. Yanenko, "Investigation of operating system influence on single event functional interrupts using fault injection and hardware error detection in ARM microcontroller," in *IEEE SIBCON*, 2021.

[42] M. V. O'Bryan, K. A. LaBel, D. Chen, M. J. Campola, M. C. Casey, J.-M. Lauenstein, J. A. Pellish, R. L. Ladbury, and M. D. Berg, "Compendium of current single event effects for candidate spacecraft electronics for NASA," in *Nuclear and Space Radiation Effects Conference (NSREC)*, 2015.

[43] C. Virmontois, A. Toulemont, G. Rolland, A. Materne, V. Lalucaa, V. Goiffon, C. Codreanu, C. Durnez, and A. Bardoux, "Radiation-induced dose and single event effects in digital CMOS image sensors," *IEEE T. Nuclear Science*, 2014.

[44] S. B. Sukhavasi, S. B. Sukhavasi, K. Elleithy, S. Abuzneid, and A. Elleithy, "CMOS image sensors in surveillance system applications," *Sensors*, 2021.

[45] G. Hopkinson, "Radiation effects in a CMOS active pixel sensor," *IEEE Transactions on Nuclear Science*, 2000.

[46] A. Huber, G. Sergienko, D. Kinna, V. Huber, A. Milocco, L. Mercadier, I. Balboa, S. Conroy, S. Cramp, V. Kiptily, U. Kruezi, H. T. Lambertz, C. Linsmeier, G. Matthews, S. Popovichev, P. Mertens, S. Silburn, and K.-D. Zastrow, "Response of the imaging cameras to hard radiation during JET operation," *Fusion Engineering and Design*, 2017.

[47] Y. Cai, L. Wen, Y. Li, Q. Guo, D. Zhou, J. Feng, X. Zhang, B. Liu, and J. Fu, "Single-Event Effects in Pinned Photodiode CMOS Image Sensors: SET and SEL," *IEEE Transactions on Nuclear Science*, 2020.

[48] Jet Propulsion Laboratory. (2023) NASA's Juno Team Assessing Camera After 48th Flyby of Jupiter. https://www.nasa.gov/missions/juno/nasas-juno-team-assessing-camera-after-48th-flyby-of-jupiter/. NASA.

[49] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *EASE*, 2014.

[50] F. C. Bruhn, N. Tsog, F. Kunkel, O. Flordal, and I. Troxel, "Enabling radiation tolerant heterogeneous GPU-based onboard data processing in space," *CEAS Space Journal*, 2020.

[51] "Connected papers," https://www.connectedpapers.com/.

[52] "European State Agency—OPS-SAT Space Lab Community Platform,"

https://opssat1.esoc.esa.int/projects/publications.
[53] F. Latorre, D. Spiller, S. Sasidharan, S. Basheer, and F. Curti, "Transfer learning for real-time crater detection on asteroids using a Fully Convolutional Neural Network," *Icarus*, 2023.
[54] L. Pauly, W. Rharbaoui, C. Shneider, A. Rathinam, V. Gaudillière, and D. Aouada, "A survey on deep learning-based monocular spacecraft pose estimation: Current state, limitations and prospects," *Acta Astronautica*, 2023.
[55] B. Haser and R. Förstner, "Reliability of Neural Networks: A Fault Injector for Space related Perturbations," *IAC*, 2022.
[56] H. Li, H. Zheng, C. Han, H. Wang, and M. Miao, "Onboard spectral and spatial cloud detection for hyperspectral remote sensing images," *Remote Sensing*, 2018.
[57] M. Pastena, B. C. Domínguez, P. P. Mathieu, A. Regan, M. Esposito, S. Conticello, C. V. Dijk, N. Vercruyssen, and P. Foglia, "ESA Earth observation directorate NewSpace initiatives," in *Small Satellite Conference*, 2019.
[58] M. Esposito, S. S. Conticello, M. Pastena, and B. C. Domínguez, "In-orbit demonstration of artificial intelligence applied to hyperspectral and thermal sensing from space," in *CubeSats and SmallSats for remote sensing III*, 2019.
[59] E. Lemaire, M. Moretti, L. Daniel, B. Miramond, P. Millet, F. Feresin, and S. Bilavarn, "An FPGA-based Hybrid Neural Network accelerator for embedded satellite image classification," in *IEEE International Symposium on Circuits and Systems*, 2020.
[60] G. Furano, G. Meoni, A. Dunne, D. Moloney, V. Ferlet-Cavrois, A. Tavoularis, J. Byrne, L. Buckley, M. Psarakis, K.-O. Voss *et al.*, "Towards the use of artificial intelligence on the edge in space systems: Challenges and opportunities," *IEEE Aerospace and Electronic Systems Magazine*, 2020.
[61] B. Denby and B. Lucia, "Orbital edge computing: Nanosatellite constellations as a new class of computer system," in *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, 2020.
[62] A. Maskey and M. Cho, "CubeSatNet: Ultralight Convolutional Neural Network designed for on-orbit binary image classification on a 1U CubeSat," *Engineering Applications of Artificial Intelligence*, 2020.
[63] P. Reiter, P. Karagiannakis, M. Ireland, S. Greenland, and L. Crockett, "FPGA acceleration of a quantized neural network for remote-sensed cloud detection," in *7th International Workshop on On-Board Payload Data Compression*, 2020.
[64] G. Mateo-Garcia, J. Veitch-Michaelis, L. Smith, S. V. Oprea, G. Schumann, Y. Gal, A. G. Baydin, and D. Backes, "Towards global flood mapping onboard low cost satellites with machine learning," *Scientific reports*, 2021.
[65] E. Rapuano, G. Meoni, T. Pacini, G. Dinelli, G. Furano, G. Giuffrida, and L. Fanucci, "An FPGA-Based Hardware Accelerator for CNNs Inference on Board Satellites: Benchmarking with Myriad 2-Based Solution for the CloudScout Case Study," *Remote Sensing*, 2021.
[66] M. P. Del Rosso, A. Sebastianelli, D. Spiller, P. P. Mathieu, and S. L. Ullo, "On-Board Volcanic Eruption Detection through CNNs and Satellite Multispectral Imagery," *Remote Sensing*, 2021.
[67] A. S. Kucik and G. Meoni, "Investigating Spiking Neural Networks for Energy-Efficient On-Board AI Applications. A Case Study in Land Cover and Land Use Classification," in *CVPR*, 2021.
[68] L. Diana, J. Xu, and L. Fanucci, "Oil spill identification from SAR images for low power embedded systems using CNN," *Remote Sensing*, 2021.
[69] L. Ferrari, F. Dell'Acqua, P. Zhang, and P. Du, "Integrating EfficientNet into an HAFNet Structure for Building Mapping in High-Resolution Optical Earth Observation Data," *Remote Sensing*, 2021.
[70] T. Pacini, E. Rapuano, G. Dinelli, and L. Fanucci, "A multi-cache system for on-chip memory optimization in FPGA-based CNN accelerators," *Electronics*, 2021.
[71] T. I. Leong, Y. M. Abbas, M. A. C. Purio, and H. A. Elmegharbel, "Image Classification Unit: A U-Net Convolutional Neural Network for On-Orbit Cloud Detection Aboard CubeSats," in *IEEE International Geoscience and Remote Sensing Symposium*, 2021.
[72] P. Fernando and J. Wei-Kocsis, "Towards a Disaster Response System Based on CubeSat Constellations," in *IEEE Cognitive Communications for Aerospace Applications Workshop*, 2021.
[73] T. Garrett and A. D. George, "Improving dependability of onboard deep learning with resilient tensorflow," in *IEEE SCC*, 2021.
[74] A. J. Farr, I. Petrunin, G. Kakareko, and J. Cappaert, "Self-supervised vessel detection from low resolution satellite imagery," in *AIAA SCITECH 2022 Forum*, 2022.
[75] V. Růžička, A. Vaughan, D. De Martini, J. Fulton, V. Salvatelli, C. Bridges, G. Mateo-Garcìa, and V. Zantedeschi, "RaVÆn: unsupervised change detection of extreme events using ML on-board satellites," *Scientific reports*, 2022.
[76] M. H. b. Azami, N. C. Orger, V. H. Schulz, T. Oshiro, and M. Cho, "Earth observation mission of a 6U CubeSat with a 5-meter resolution for wildfire image classification using convolution neural network approach," *Remote Sensing*, 2022.
[77] C. Luo, S. Feng, X. Yang, Y. Ye, X. Li, B. Zhang, Z. Chen, and Y. Quan, "LWCDnet: A Lightweight Network for Efficient Cloud Detection in Remote Sensing Images," *IEEE Transactions on Geoscience and Remote Sensing*, 2022.
[78] G. Labrèche, D. Evans, D. Marszk, T. Mladenov, V. Shiradhonkar, T. Soto, and V. Zelenevskiy, "OPS-SAT Spacecraft Autonomy with TensorFlow Lite, Unsupervised Learning, and Online Machine Learning," in *IEEE Aerospace Conference*. IEEE, 2022.
[79] D. Spiller, K. Thangavel, S. T. Sasidharan, S. Amici, L. Ansalone, and R. Sabatini, "Wildfire segmentation analysis from edge computing for on-board real-time alerts using hyperspectral imagery," in *IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering*, 2022.
[80] C. Luo, S. Feng, X. Li, Y. Ye, B. Zhang, Z. Chen, and Y. Quan, "ECDNet: A bilateral lightweight cloud detection network for remote sensing images," *Pattern Recognition*, 2022.
[81] C. Salazar, J. Gonzalez-Llorente, L. Cardenas, J. Mendez, S. Rincon, J. Rodriguez-Ferreira, and I. F. Acero, "Cloud detection autonomous system based on machine learning and cots components on-board small satellites," *Remote Sensing*, 2022.
[82] R. Pitonak, J. Mucha, L. Dobis, M. Javorka, and M. Marusin, "CloudSatNet-1: FPGA-based hardware-accelerated quantized CNN for satellite on-Board cloud coverage classification," *Remote Sensing*, 2022.
[83] G. Guerrisi, F. Del Frate, and G. Schiavon, "Satellite On-Board Change Detection via Auto-Associative Neural Networks," *Remote Sensing*, 2022.
[84] M. Jeon, T. Kim, C. Lee, and C.-H. Youn, "A Channel Pruning Optimization With Layer-Wise Sensitivity in a Single-Shot Manner Under Computational Constraints," *IEEE Access*, 2022.
[85] D. A. Zeleke and H.-D. Kim, "A New Strategy of Satellite Autonomy with Machine Learning for Efficient Resource Utilization of a Standard Performance CubeSat," *Aerospace*, 2023.
[86] M. P. Del Rosso, A. Sebastianelli, D. Spiller, and S. L. Ullo, "A demo setup testing onboard CNNs for Volcanic Eruption Detection," in *IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering (MetroXRAINE)*, 2022.
[87] J. Murphy, J. E. Ward, and B. M. Namee, "Developing Machine Learning Models for Space Based Edge AI Platforms," in *Small Satellite Conference*, 2022.
[88] D. Spiller, A. Carbone, F. Latorre, and F. Curti, "Hardware-in-the-loop Simulations of Remote Sensing Disaster Monitoring Systems with Real-Time On-Board Computation," in *IEEE International Conference on Metrology for Extended Reality, Artificial Intelligence and Neural Engineering*, 2022.
[89] L. Buckley, A. Dunne, G. Furano, and M. Tali, "Radiation Test and in Orbit Performance of MpSoC AI Accelerator," in *IEEE Aerospace Conference*, 2022.
[90] G. Mateo-García, J. Veitch-Michaelis, C. Purcell, N. Longepe, S. Reid, A. Anlind, F. Bruhn, J. Parr, and P. P. Mathieu, "In-orbit demonstration of a re-trainable machine learning payload for processing optical imagery," *Scientific Reports*, 2023.
[91] G. Labrèche, D. Evans, D. Marszk, T. Mladenov, V. Shiradhonkar, and V. Zelenevskiy, "Artificial Intelligence for Autonomous Planning and Scheduling of Image Acquisition with the SmartCam App On-Board the OPS-SAT Spacecraft," in *AIAA SCITECH 2022 Forum*, 2022.
[92] S. Kacker, A. Meredith, K. Cahoy, and G. Labreche, "Machine learning image processing algorithms onboard OPS-SAT," in *Small Satellite Conference*, 2022.
[93] T. Mladenov, G. Labreche, T. Syndercombe, and D. Evans, "Augmenting Digital Signal Processing with Machine Learning techniques using the Software Defined Radio on the OPS-SAT Space Lab," in *International Astronautical Congress*, 2022.
[94] E. Kervennic, T. Louis, M. Benguigui, F. Férésin, Y. Bobichon, and

A. Girard, "Deployment of a cloud segmentation neural network on embedded hardware targets and benchmark of the different deployment toolchains." in *International Workshop on On-Board Payload Data Compression*, 2022.
[95] S. Fratini, N. Policella, R. Silva, and J. Guerreiro, "On-board autonomy operations for OPS-SAT experiment," *Applied Intelligence*, 2022.
[96] L. Gu, Q. Fang, Z. Wang, E. Popov, and G. Dong, "Learning lightweight and superior detectors with feature distillation for onboard remote sensing object detection," *Remote Sensing*, 2023.
[97] G. Guerrisi, F. Del Frate, and G. Schiavon, "Artificial Intelligence based on-board image compression for the Φ-Sat-2 mission," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2023.
[98] G. Cascelli, C. Guaragnella, R. Nutricato, K. Tijani, A. Morea, N. Ricciardi, and D. O. Nitti, "Use of a Residual Neural Network to Demonstrate Feasibility of Ship Detection Based on Synthetic Aperture Radar Raw Data," *Technologies*, 2023.
[99] M. Coca and M. Datcu, "FPGA Accelerator for Meta-Recognition Anomaly Detection: Case of Burned Area Detection," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2023.
[100] Q. Shi, L. Li, J. Feng, W. Chen, and J. Yu, "Automated Model Hardening with Reinforcement Learning for On-Orbit Object Detectors with Convolutional Neural Networks," *Aerospace*, 2023.
[101] C. Serief, Y. Ghelamallah, and Y. Bentoutou, "Deep Learning-based System for Change Detection On-Board Earth Observation Small Satellites," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 2023.
[102] C. Kadway, S. Dey, A. Mukherjee, A. Pal, and G. Bézard, "Low Power & Low Latency Cloud Cover Detection in Small Satellites Using On-board Neuromorphic Processors," in *International Joint Conference on Neural Networks (IJCNN)*, 2023.
[103] N. Ferrante, G. Giuffrida, P. Nannipieri, A. Bechini, and L. Fanucci, "Fault Detection Exploiting Artificial Intelligence in Satellite Systems," in *International Conference on Applied Intelligence and Informatics*, 2022.
[104] A. Carbone, D. Spiller, S. Amici, K. Thangavel, R. Sabatini, and G. Laneve, "Comparison of 1D and 3D Convolutional Neural Networks for Wildfire Detection Using PRISMA Hyperspectral Imagery and Domain Adaptation," in *IEEE International Conference on Metrology for eXtended Reality, Artificial Intelligence and Neural Engineering*, 2023.
[105] R. Ciardi, G. Giuffrida, G. Benelli, C. Cardenio, and R. Maderna, "GPU@SAT: A General-Purpose Programmable Accelerator for on Board Data Processing and Satellite Autonomy," in *International Conference on Applied Intelligence and Informatics*, 2022.
[106] R. Deticio, A. Bandala, J. A. Jose, R. Concepcion II, M. A. Purio, E. Sybingco, and R. J. T. Ai, "Improving the U-Net Segmentation Model for Land Cover Classification in Satellite Image Processing," in *IEEE Region 10 Conference (TENCON)*, 2023.
[107] R. Deticio, A. Bandala, J. A. Jose, R. C. Ii, M. A. Purio, E. Sybingco, and R. J. T. Ai, "Application of a U-Net Segmentation Model in Land Cover Classification for Use in Automated Data Prefiltering Onboard Nanosatellites," in *IEEE Region 10 Conference*, 2023.
[108] V. Leon, P. Minaidis, G. Lentaris, and D. Soudris, "Accelerating AI and Computer Vision for Satellite Pose Estimation on the Intel Myriad X Embedded SoC," *Microprocessors and Microsystems*, 2023.
[109] T. Fernando, C. Fookes, H. Gammulle, S. Denman, and S. Sridharan, "Towards On-Board Panoptic Segmentation of Multispectral Satellite Images," *IEEE Transactions on Geoscience and Remote Sensing*, 2023.
[110] J. Murphy, J. E. Ward, and B. Mac Namee, "An Overview of Machine Learning Techniques for Onboard Anomaly Detection in Satellite Telemetry," in *European Data Handling & Data Processing Conference (EDHPC)*, 2023.
[111] J. Nalepa, B. Ruszczak, K. Kotowski, J. Andrzejewski, A. Musiał, D. Evans, V. Zelenevskiy, S. Bammens, and R. Laurinovičs, "Look ma, no ground truth! On building supervised anomaly detection from OPS-SAT telemetry," in *International Astronautical Congress*, 2023.
[112] P. Miralles, K. Thangavel, A. F. Scannapieco, N. Jagadam, P. Baranwal, B. Faldu, R. Abhang, S. Bhatia, S. Bonnart, I. Bhatnagar *et al.*, "A critical review on the state-of-the-art and future prospects of Machine Learning for Earth Observation Operations," *Advances in Space Research*, 2023.
[113] F. Libano, B. Wilson, J. Anderson, M. J. Wirthlin, C. Cazzaniga, C. Frost, and P. Rech, "Selective hardening for neural networks in FPGAs," *IEEE Transactions on Nuclear Science*, 2018.
[114] K. Adam, I. Mohamed, and Y. Ibrahim, "A Selective Mitigation Technique of Soft Errors for DNN Models Used in Healthcare Applications: DenseNet201 Case Study," *IEEE Access*, 2021.
[115] Y. Ibrahim, H. Wang, M. Bai, Z. Liu, J. Wang, Z. Yang, and Z. Chen, "Soft Error Resilience of Deep Residual Networks for Object Recognition," *IEEE Access*, 2020.
[116] S. Sabogal, A. George, and G. Crum, "ReCoN: A reconfigurable CNN acceleration framework for hybrid semantic segmentation on hybrid SoCs for space applications," in *IEEE Space Computing Conference*, 2019.
[117] "Satellitegithub," https://github.com/satellite-image-deep-learning.
[118] S. Mohajerani and P. Saeedi, "Cloud-net+: A cloud segmentation cnn for landsat 8 remote sensing imagery optimized with filtered jaccard loss function," *arXiv*, 2020.
[119] J. Timpe, K. O'Neill, D. Qendri, B. Berkane, G. Chapman, and D. Quinn, "Application of AMD Versal™ Adaptive SoC to Radar Space Time Adaptive Processing in Space," in *European Data Handling & Data Processing Conference*. IEEE, 2023.
[120] A. Costin, H. Turtiainen, S. Khandker, and T. Hämäläinen, "Towards a Unified Cybersecurity Testing Lab for Satellite, Aerospace, Avionics, Maritime, Drone (SAAMD) technologies and communications," in *SpaceSec (NDSS-W)*, 2023.
[121] "NVBitFI," https://github.com/NVlabs/nvbitfi.
[122] T. Tsai, S. K. S. Hari, M. Sullivan, O. Villa, and S. W. Keckler, "NVBitFI: Dynamic Fault Injection for GPUs," in *IEEE/IFIP Int. Conf. Depend. Syst. Netw.*, 2021.
[123] "LLTFI," https://github.com/DependableSystemsLab/LLTFI.
[124] "Kornia," https://github.com/kornia/kornia.
[125] E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. Bradski, "Kornia: an Open Source Differentiable Computer Vision Library for PyTorch," in *IEEE Winter Conf. Appl. Comp. Vis.*, 2020.
[126] A. Mahmoud, N. Aggarwal, A. Nobbe, J. R. S. Vicarte, S. V. Adve, C. W. Fletcher, I. Frosio, and S. K. S. Hari, "PyTorchFI: A Runtime Perturbation Tool for DNNs," in *IEEE/IFIP Int. Conf. Dependable Systems Networks Workshops*, 2020.
[127] Z. Chen, N. Narayanan, B. Fang, G. Li, K. Pattabiraman, and N. DeBardeleben, "TensorFI: A Flexible Fault Injection Framework for TensorFlow Applications," in *IEEE ISSRE*, 2020.
[128] F. F. Dos Santos, A. Kritikakou, J. E. R. Condia, J.-D. Guerrero-Balaguera, M. S. Reorda, O. Sentieys, and P. Rech, "Characterizing a neutron-induced fault model for deep neural networks," *IEEE Transactions on Nuclear Science*, 2023.
[129] S. Mohajerani and P. Saeedi, "Cloud-Net: An end-to-end cloud detection algorithm for Landsat 8 imagery," in *IEEE IGARSS*, 2019.
[130] "ONNX-MLIR-LLTFI repository," https://github.com/DependableSystemsLab/onnx-mlir-lltfi/tree/LLTFI.
[131] H. Guo, H. Bai, and W. Qin, "ClouDet: A dilated separable CNN-based cloud detection framework for remote sensing imagery," *IEEE J. Sel. Topics Applied Earth Obs. Remote Sens.*, 2021.
[132] "ONNX-MLIR repository," https://github.com/onnx/onnx-mlir.
[133] K. A. Kalpoma, A. S. Aurgho, M. M. I. Shizan, F. H. Ani, and A. R. Bondhon, "Deep learning image segmentation for satellite images of national highways of Bangladesh," in *IEEE IGARSS*, 2023.
[134] J. Su, D. V. Vargas, and K. Sakurai, "One pixel attack for fooling deep neural networks," *IEEE Transactions on Evolutionary Computation*, 2019.
[135] G. Apruzzese, H. S. Anderson, S. Dambra, D. Freeman, F. Pierazzi, and K. Roundy, ""Real Attackers Don't Compute Gradients": Bridging the Gap Between Adversarial ML Research and Practice," in *SaTML*, 2023.
[136] N. Abderrahmane, B. Miramond, E. Kervennic, and A. Girard, "SPLEAT: SPiking Low-power Event-based ArchiTecture for in-orbit processing of satellite imagery," in *International Joint Conference on Neural Networks (IJCNN)*, 2022.
[137] M. J. Veyette, K. Aylor, D. Stafford, M. Herrera, S. Jumani, C. Lineberry, C. Macklen, E. Maxwell, R. Stiles, and M. Jenkins, "Ai/ml for mission processing onboard satellites," in *AIAA SCITECH 2022 Forum*, 2022.
[138] M. Esposito, S. Conticello, M. Pastena, and B. Carnicero Domínguez, "Hyperscout-2: Highly integration of hyperspectral and thermal sensing for breakthrough in-space applications," *Proceedings of the ESA Earth Observation φ-Week*, 2019.

[139] M. Esposito, B. Dominguez, M. Pastena, N. Vercruyssen, S. Conticello, C. van Dijk, P. Manzillo, and R. Koeleman, "Highly integration of hyperspectral, thermal and artificial intelligence for the ESA PHISAT-1 mission," in *Proceedings of the International Astronautical Congress IAC*, 2019.

[140] L. Chavier, B. Bonham-Carter, H. Burd, T. Heydrich, G. O'Shea, J. Prud'homme, N. Ayyappan, M. Maharib, T. Ganesalingam, A. Higginson *et al.*, "Deploying Artificial Intelligence Capabilities by Hybridizing a Neural Network on a Satellite," in *ASCEND 2023*, 2023.

[141] B. Ruszczak, K. Kotowski, J. Andrzejewski, A. Musiał, D. Evans, V. Zelenevskiy, S. Bammens, R. Laurinovics, and J. Nalepa, "Machine Learning Detects Anomalies in OPS-SAT Telemetry," in *International Conference on Computational Science*, 2023.

[142] V. Fanizza, D. Rijlaarsdam, P. T. T. González, and J. L. Espinosa-Aranda, "Transfer Learning for On-Orbit Ship Segmentation," in *European Conference on Computer Vision*, 2022.

[143] A. J. Macdonald, A. Budhkar, B. Bonham-Carter, E. Smal, M. Cross, K. V. Raimalwala, M. Battler, and M. Faragalli, "Enabling Autonomy with a Deep Learning Framework for Planetary Exploration," in *ASCEND*, 2022.

[144] C. Wilson, S. Sabogal, A. George, and A. Gordon-Ross, "Hybrid, adaptive, and reconfigurable fault tolerance," in *IEEE Aerospace Conference*, 2017.

[145] S. Sabogal and A. George, "A Methodology for Evaluating and Analyzing FPGA-Accelerated, Deep-Learning Applications for Onboard Space Processing," in *IEEE Space Computing Conference*, 2021.

[146] E. T. Kain, T. M. Lovelly, and A. D. George, "Evaluating SEU resilience of CNNs with fault injection," in *IEEE High Performance Extreme Computing Conference*, 2020.

[147] L. M. Luza, A. Ruospo, D. Söderström, C. Cazzaniga, M. Kastriotou, E. Sanchez, A. Bosio, and L. Dilillo, "Emulating the effects of radiation-induced soft-errors for the reliability assessment of neural networks," *IEEE Transactions on Emerging Topics in Computing*, 2021.

[148] D. Olszewski, A. Lu, C. Stillman, K. Warren, C. Kitroser, A. Pascual, D. Ukirde, K. Butler, and P. Traynor, ""Get in Researchers; We're Measuring Reproducibility": A Reproducibility Study of Machine Learning Papers in Tier 1 Security Conferences," in *ACM CCS*, 2023.

[149] A. Ding, M. Chan, A. Hass, N. O. Tippenhauer, S. Ma, and S. Zonouz, "Get Your Cyber-Physical Tests Done! Data-Driven Vulnerability Assessment of Robotic Aerial Vehicles," in *IEEE DSN*, 2023.

[150] M. Strohmeier, I. Martinovic, and V. Lenders, "A k-NN-based localization approach for crowdsourced air traffic communication networks," *IEEE Transactions on Aerospace and Electronic Systems*, 2018.

[151] R. P. Dick, R. Aitken, J. Mogill, J. P. Strachan, K. Bresniker, W. Lu, Y. Nakahira, Z. Li, M. J. Marinella, W. Severa *et al.*, "Research challenges for energy-efficient computing in automated vehicles," *Computer*, 2023.

[152] N. Xue, L. Niu, X. Hong, Z. Li, L. Hoffaeller, and C. Pöpper, "DeepSIM: GPS Spoofing Detection on UAVs using Satellite Imagery Matching," in *ACSAC*, 2020.

[153] N. Martin, T. Cook, A. George, and B. M. Grainger, "Radiation-Tolerant, High-Power Density GaN Drop-On Point-of-Load Converters," in *IEEE Aerospace Conference*, 2022.

[154] N. Perryman, C. Wilson, and A. George, "Evaluation of Xilinx Versal Architecture for Next-Gen Edge Computing in Space," in *IEEE Aerospace Conference*, 2023.

[155] N. Franconi, T. Cook, C. Wilson, and A. D. George, "Comparison of multi-phase power converters and power delivery networks for next-generation space architectures," in *IEEE Aerospace Conference*, 2023.

[156] J. R. Kocik and A. D. George, "Space Station Power Forecasting with LSTMs for an Embedded Platform," in *IEEE National Aerospace and Electronics Conference*, 2021.

[157] J. Dong, H. Qiu, Y. Li, T. Zhang, Y. Li, Z. Lai, C. Zhang, and S.-T. Xia, "One-bit flip is all you need: When bit-flip attack meets model training," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023.

[158] M. Lin, M. Cheng, D. Luo, and Y. Chen, "CLExtract: Recovering Highly Corrupted DVB/GSE Satellite Stream with Contrastive Learning," *SpaceSec (NDSS-W)*, 2023.

[159] A. Du, Y. W. Law, M. Sasdelli, B. Chen, K. Clarke, M. Brown, and T.-J. Chin, "Adversarial attacks against a satellite-borne multispectral cloud detector," in *International Conference on Digital Image Computing: Techniques and Applications*, 2022.

[160] J. Willbold, M. Schloegel, M. Vögele, M. Gerhardt, T. Holz, and A. Abbasi, "Space Odyssey: An Experimental Software Security Analysis of Satellites," in *IEEE S&P*, 2023.

[161] D. Fischer, I. Aguilar Sanchez, B. Saba, G. Moury, B. Bailey, C. Biggerstaff, H. Weiss, M. Pilgram, and D. Richter, "Finalizing the CCSDS space-data link layer security protocol: Setup and execution of the interoperability testing," in *AIAA SPACE Conference and Exposition*, 2015.

[162] M. Adalier, A. Riffel, M. Galvan, B. Johnson, and S. Burleigh, "Efficient and secure autonomous communications for deep space missions," in *IEEE Aerospace Conference*, 2020.

[163] G. Falco, R. Thummala, and A. Kubadia, "Wannafly: An approach to satellite ransomware," in *IEEE SMC-IT*, 2023.

[164] G. Giuliari, T. Ciussani, A. Perrig, and A. Singla, "{ICARUS}: Attacking low earth orbit satellite networks," in *USENIX ATC*, 2021.

[165] J. Smailes, S. Köhler, S. Birnbach, M. Strohmeier, and I. Martinovic, "Watch this space: Securing satellite communication through resilient transmitter fingerprinting," in *ACM CCS*, 2023.