Onboard AI for Fire Smoke Detection Using Hyperspectral Imagery: An Emulation for the Upcoming Kanyini Hyperscout-2 Mission

Sha Lu, Eriita Jones, Liang Zhao, Yu Sun, Kai Qin, *Senior Member, IEEE*, Jixue Liu, Jiuyong Li, *Member, IEEE*, Prabath Abeysekara, Norman Mueller, Simon Oliver, Jim O'Hehir, and Stefan Peters

***Abstract*—This article presents our research in the prelaunch phase of the Kanyini mission, which aims to implement an energy-efficient, AI-based system onboard for early fire smoke detection using hyperspectral imagery. Our approach includes three key components: developing a diverse hyperspectral training dataset from VIIRS imagery, groundwork in band selection and AI model preparation, and developing an emulation system. We adapted and evaluated our previously developed lightweight convolutional neural network model, VIB_SD, to meet the computational constraints of satellite deployment. The emulation system tests various onboard AI tasks and processes. Our comprehensive experiments demonstrate the feasibility and benefits of employing onboard AI for fire smoke detection, significantly improving downlink efficiency, energy consumption, and detection speed.**

***Index Terms*—Artificial intelligence (AI) onboard, fire smoke detection, satellite emulation.**

I. Introduction

With the escalating threat of wildfires due to climate change, the need for early detection has become paramount in minimizing their destructive impact on society, ecosystems, and economy [1], [2]. Satellite remote sensing has emerged as a cost-effective and reliable tool for fire detection, benefiting from the growing deployment of satellites dedicated to Earth monitoring [3], [4], [5], [6]. As smoke is usually the first thing you can see from space before the fire gets hot and big enough for sensors to detect fire heat, detecting fire smoke becomes crucial for early warning and timely response to mitigate potential risks and damages.

However, the prevalence of microsats and nanosats and the increased spatial and spectral resolution of imagery captured by modern Earth observation sensors have greatly increased bandwidth usage. This has led to research into optimizing up/downlink bandwidth resources. For many sensor systems, only a fraction of the data collected contains critical information related to the specific purpose of a mission. To address this issue, recent advances in low-power computing platforms and the advent of artificial intelligence (AI) technology have paved the way for the adoption of edge computing [7]. By leveraging hardware accelerators and deploying efficient algorithms, such as convolution neural networks (CNNs) onboard, tasks such as early fire smoke detection can be performed, allowing for timely alarm generation in the event of bushfires [8], [9], [10].

The Kanyini satellite mission [11] is a collaborative effort between the South Australian Government, the SmartSat Cooperative Research Centre, and industry partners, including Inovor Technologies and Myriota. The mission aims to launch a 6 U CubeSat satellite into low Earth orbit to collect data on bushfire preparedness, response, and resilience, as well as inland and coastal water quality. Equipped with a hyperspectral imager [HyperScout-2 (HS2) manufactured by Cosine], the satellite sensor will capture reflected light from Earth in different wavelengths to generate detailed surface maps for various applications, including bushfire monitoring, water quality assessment, and land management. The anticipated launch year is 2024, with an estimated cost of $6.5 million. The collected data will be publicly accessible and utilized by government agencies, businesses, and researchers. This mission holds significance for the space industry of South Australia, being the first state-based satellite in Australia, and is expected to contribute to the state–space sector growth. Moreover, the acquired data will aid in enhancing bushfire and water resource management within South Australia.

Our research project, part of the Kanyini mission, aims to provide a solution for energy-efficient AI-based onboard processing of hyperspectral imagery for early fire smoke detection. This work fills the gap of using a small AI model to detect fire smokes in a cube sat with limited computation and data downlinking capabilities. To the best of our knowledge, the work [12] and [13] is the only one similar in settings to ours but it is in the area of cloud detection. Our work is different in the following aspects: our prelaunch study does not have available hyperspectral dataset matching the bandwidths of the sensor to be launched; and our work is for newer generation of

Manuscript received 21 December 2023; revised 24 March 2024; accepted 24 April 2024. Date of publication 29 April 2024; date of current version 14 May 2024. This work was supported by the SmartSat CRC, whose activities are funded by the Australian Government's CRC Program. *(Corresponding author: Stefan Peters.)*

Sha Lu, Liang Zhao, Jixue Liu, Jiuyong Li, Jim O'Hehir, and Stefan Peters are with the University of South Australia, Adelaide, SA 5001, Australia (e-mail: stefan.peters@unisa.edu.au).

Eriita Jones is with the Curtin University, Bentley, WA 6102, Australia.

Yu Sun, Kai Qin, and Prabath Abeysekara are with the Swinburne University of Technology, Hawthorn, VIC 3122, Australia.

Norman Mueller and Simon Oliver are with the Geoscience Australia, Symonston, ACT 2609, Australia.

Digital Object Identifier 10.1109/JSTARS.2024.3394574

© 2024 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/

VPUs whereby the development environment has been updated and some functions available in the version used in Esposito et al.'s [12] work are not available in the new version. We report detailed hardware, software settings, and the performances of not-only smoke model accuracy, but also results about bands selection, and downlinking data sizes.

This project consists of two phases: prelaunch and postlaunch. The prelaunch phase focuses on evaluating the feasibility and benefits of onboard smoke detection, while the postlaunch phase deploys and optimizes the AI-based fire smoke detection system to Kanyini, leveraging real satellite data and refining onboard processing capabilities to enhance accuracy and efficiency. This article presents our work in the prelaunch phase.

To evaluate the feasibility of onboard smoke detection, we have chosen our previously developed CNN-based model, variant input bands for smoke detection (VIB_SD) [14], as a suitable lightweight AI model for the Kanyini/HS2 mission. VIB_SD was originally designed and trained with Landsat-8 imagery, it has not yet been tested for onboard tasks and was trained with different number of bands to classify a different set of classes. Therefore, it requires some adaptations, alongside retraining, and comprehensive evaluation, to ensure its effective deployment on the Kanyini satellite.

Since the actual computing environment and real hyperspectral imagery captured by HS2 onboard the Kanyini satellite were not available during the early stage of the mission, we simulated a comprehensive training dataset from VIIRS imagery, which encompassed a wide range of generated hyperspectral imagery and the dataset covers various fire smoke scenarios across Australia. To evaluate the onboard performance, we developed an emulation system to evaluate each step of onboard processes and AI tasks using computational capabilities similar to those of the Kanyini satellite.

Our experiments have demonstrated significant advantages of integrating AI onboard for smoke detection in satellite imagery, as compared to traditional methods. As detailed in Section IV-C, for scenarios where 10% of the imagery contains fire smoke, the AI onboard approach markedly reduces the data downlink volume to just 16% of its original size (from 388 to 61 MB), resulting in an 84% decrease in energy consumption (from 0.414 to 0.065 WH). Such efficiency is particularly advantageous considering the limited downlinking capacity during each satellite pass, underscoring the effectiveness of onboard AI in optimizing resource usage in satellite-based monitoring tasks.

In summary, the major contributions of our work presented in this article are as follows.

1) *Generation of a comprehensive hyperspectral training dataset:* We created a training dataset using the visible infrared imaging radiometer suite (VIIRS) imagery that encompasses diverse variations of fire smoke. The dataset includes four fire smoke scene-related classes, namely "Smoke," "Cloud," "Mixed," and "Clear." This dataset provides a valuable resource for training and evaluating fire smoke detection models.
2) *Emulation system for onboard performance evaluation:* We developed an emulation system to evaluate the onboard performance of different processes and AI tasks. This system serves as a valuable tool for optimizing algorithms and workflows before the deployment on actual satellite systems, ensuring the efficiency and effectiveness of onboard processing.
3) *Adaptation and deployment of the VIB_SD model:* We adapted the VIB_SD model, a lightweight CNN-based approach suitable to operate within the computational and data transfer constraints of the HS2 sensor on the Kanyini satellite. The VIB_SD model demonstrates a high prediction accuracy and achieves a low false negative rate (FNR) on the simulated dataset, indicating its effectiveness in detecting fire smoke.
4) *Comprehensive experimental evaluation:* We conducted comprehensive experiments to evaluate the various scenarios of onboard processing, providing empirical evidence on the feasibility and significance of AI onboard smoke detection. These experiments validate the benefits of onboard smoke detection in terms of downlink efficiency, energy consumption, and detection speed.

The rest of this article is organized as follows. Section II reviews the related work in AI onboard and fire/smoke detection. in Section III, we provide an overview of the simulation process for the training dataset, describe the architecture of VIB_SD, and detail the design of the emulation system. Section IV presents and discusses the experiments and results on the prediction accuracy of VIB_SD with different selected bands and the emulation results of various onboard processes. Section IV-D discusses the limitations and constraints of the research. Finally, Section V concludes this article.

II. Related Work

The advancement of AI in satellite systems has been marked by substantial progress. Initially, research in this area was concentrated on the application of AI algorithms and machine learning techniques for on-ground processing of satellite imagery data. These applications span a range of fields including land cover classification [15], [16], vegetation monitoring [17], [18], water quality assessment [19], disaster response [20], climate change monitoring [21], [22], and smoke detection [5], [6].

More recently, the focus has shifted toward the onboard deployment of AI in satellite systems. Implementing AI directly on satellites offers several distinct advantages such as autonomous decision-making and real-time data analysis, thereby significantly improving mission performance [23]. However, deploying AI onboard also presents unique challenges, particularly constraints related to hardware and computational capacity [8], [23].

To mitigate these constraints, several studies have explored AI solutions in the context of onboard optical and multispectral imaging. Salazar et al. [24] proposed a CNN to autonomously prioritize RGB images based on cloud coverage levels, thereby optimizing the limited bandwidth of small satellites. Similarly,

Del Rosso et al. [25] focused on detecting volcanic eruptions through CNN designed specifically to work within the computational constraints of aerospace applications. Mateo-Garcia et al. [26] proposed "WorldFloods," an in-orbit retrainable machine learning payload that not only processes optical imagery for flood mapping but is also capable of "on-the-fly" model updates. Moreover, work in multispectral sensing for environmental monitoring has shown the capability of machine learning algorithms to estimate variables, such as sea ice concentration and soil moisture [27], [28].

A specific focus within these developments is the use of AI for hyperspectral imaging [12], [13], [29]. Hyperspectral data offer richer informational content but at the cost of generating larger datasets, which are challenging both to process onboard and to transmit back to Earth. One notable study published in [12] and [13] presents the first in-orbit demonstration of AI applied to hyperspectral and thermal sensing using the HS2 imager. Deployed in the ϕ-Sat-1 mission, it utilized a CNN for cloud detection with an 95% accuracy. Esposito and Marchi's [30] work showcases the in-orbit demonstration of HyperScout-1, the first hyperspectral imager for nanosatellites, with potential applications in land cover classification, vegetation monitoring, water quality assessment, and disaster response. AI has also been applied for wildfire detection in the hyperspectral domain. Thangavel et al. [31] conducted a case study using CNNs to detect wildfires autonomously in hyperspectral imagery. Further, Spiller et al. [32] investigated edge computing approaches to perform wildfire segmentation analysis directly from satellite platforms, discussing the feasibility of implementing CNNs on various hardware accelerators, such as Intel Movidius Myriad 2 and Nvidia Jetson series for real-time alerting. These advancements not only pave the way for more real-time applications but also demonstrate the growing emphasis on using AI for complex hyperspectral data analysis in an onboard setting.

With the promising benefits of current and upcoming CubeSat missions equipped with hyperspectral sensors, a thorough investigation of the energy consumption associated with application-specific AI onboard processing solutions during the prelaunch phase is crucial. Such an emulation is essential for accurate estimations of onboard performance and allows for the efficient calibration of AI models postlaunch once real-world data are available. This article focuses on the prelaunch phase.

III. Materials and Methods

In this section, we present the procedures undertaken during the prelaunch phase for AI onboard fire smoke detection, as illustrated in Fig. 1. The workflow is divided into three main components: 1) training dataset generation; 2) on-ground preparation; and 3) onboard emulation. The training dataset generation component involves the synthesis of HS2 imagery. From this synthesized data, we generate and prelabel a training dataset over a wide range of fire smoke conditions, further detailed in Section III-A. The second component, on-ground preparation, encompasses a variety of tasks including band selection analysis, AI model preparation, and the setup and tuning of the emulation system. The details are elaborated in Sections III-B, III-C, and III-D, respectively. The onboard emulation component includes the emulation of onboard performance and the evaluation of the AI model in an onboard setting. The details are presented in Section III-D.

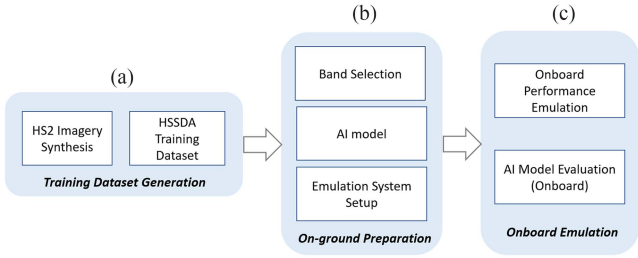

Fig. 1. Overview of the prelaunch phase for AI onboard fire smoke detection. There are three main components of the workflow. (a) Training dataset generation, where HS2 imagery is synthesized to create a prelabeled training dataset covering a diverse range of fire smoke conditions. (b) On-ground preparation, which involves tasks such as band selection analysis, AI model preparation, and setting up and tuning the emulation system. (c) Onboard emulation, depicting the emulation of onboard performance and the evaluation of the AI model in an onboard setting.

A. Training Dataset Generation

Due to the fact that the Kanyini satellite is yet to be launched, a simulated imagery dataset was generated for training the VIB_SD learning model through a series of steps outlined as follows.

1) Generating HS2 Imagery: The HS2 hyperspectral imager captures 45 spectral bands across the 400–1250 nanometre range and three bands at thermal wavelengths between 8–14 μm. HS2 achieves a ground-sampling distance of 75 m/pixel at an orbit of 500 m altitude [13], and an optimal resolution of 390 m in the thermal infrared (TIR). The NASA/NOAA Suomi NPP satellite carrying VIIRS was chosen to provide spectral data for hyperspectral imagery simulation instead as it captures daily imagery over the Australian continent. VIIRS however has a coarser resolution of at best 375 m for visible—SWIR, and 1000 m for TIR. It also has a much coarser spectral resolution being a multispectral instead of hyperspectral camera, and has only 11 bands that provide a wavelength overlap to those of HS2. Out of 48 spectral bands, this then left 27 HS2 spectral bands that required simulation. It should be noted that VIIRS imagery is captured with a radiometric resolution of 12 bits, in contrast to the 16-bit resolution of HS2 [9]. Further information can be found in Stefan et al.'s [33] work.

The VIIIRS products utilised for simulating HS2 imagery were the VNP09GA daily surface reflectance L2 product at 500 m and 1 km resolution, and the VNP09CMG daily L3 product at 0.05 ° spatial resolution that provides surface emission and brightness temperature. Both data products are corrected for atmospheric conditions. The VIIRS reflectance imagery (visible-SWIR wavelengths) was accessed via the Land Processes Distributed Active Archive Center (LP DAAC) APPEEARS data access tool [34] while the emission imagery (thermal wavelengths) was accessed via direct download link available from LP DAAC.

HS2 simulated images were produced by first transform the VIIRS imagery to the best expected resolution of HS2 image resolution, using bilinear interpolation algorithm. This does not infer any new spatial information, but merely upsamples the image pixel grid to a higher resolution. Second bandgaps or missing spectral information, of which there are 27, were filled through a combination of i) duplicating spectral information, where broader wavelength range VIIRS bands encompassed two or more HS2 bands; and ii) applying spectrally averaging of two VIIRS bands at a longer and shorter wavelength than the missing wavelength band. Poissonian distributed random noise R(λ) was multiplicatively added to each pseudoband as appropriate for electromagnetic imagery [e.g., photon noise, (Syed et al. [35])].

2) Training Sample Acquisition: They are necessary for the development of an accurate and smoke-biased detection model, the proposed training dataset imagery acquisition followed three principles.

a) *Aim to distinguish four classes:* smoke, cloud, mixtures of aerosols, and with the remaining pixels being assigned to background land-cover and land-use.
b) Aim to capture as much variance within each of those four classes so that the model is robust in a real-life scenario.
c) Encompass as much generic variations in the image data related to the HS2 satellite model, so that the model is also robust to the variations in changes to image acquisition (e.g., scale, lighting changes, season, contrast, image noise).

To produce the simulated imagery dataset, six regions of interest were chosen across five Australian states and territories, encompassing over 53 million hectares of land surface area. Image dates were chosen to overlap historical fire events in each state, resulting in the processing of over 500 VIIRS satellite images over 200 dates, and bushfire events over three years from 2018 to 2020. This broad range in spatial and temporal characteristics in the training data is essential as it allows capturing different seasonal bushfire events, occurring in different climactic zones, with imagery captured at different times of day, and viewing angles. This heterogeneity is crucial for the robustness of an AI smoke detection model.

Imagery tiles were produced by partitioning the simulated HS2 imagery (derived by methods described previously from the VIIRS satellite images), at 75 m/pixel resolution, 48 spectral bands, and one additional band consisting of a "smoke," "cloud," "smoke cloud mixtures" mask derived from the Hyperscout-2 smoke detection algorithm (HSSDA), into square grids containing 256 $\times$ 256 pixels (in the *x*-*y* dimensions). Each tile covers a region of 19.2 $\times$ 19.2 km (36 864 ha) on the ground. The square grids, or tiles, were arranged so that they had 50% overlap in both the *x* and *y* direction. Tiles were only retained if all 256 $\times$ 256 pixels in each dimension contained valid spectral information. The final neural network training dataset comprised 189 964 labeled hyperspectral tiles with classes differentiating "smoke," "cloud," and "mixtures of smoke, haze, cloud, and other aerosols" for the specific purpose of training an AI smoke detection model for hyperspectral data.

The HSSDA was developed using a risk-adverse approach, broadly adapted from Lu et al.'s [36] work and refined via empirical research whereby pixels very unlikely belonging to one of the following three aerosol classes of interest were removed. The formula for the HSSDA, and the method for assigning pixels to classes, is as follows.

1) If (NDVI $\leq$ 0.51) or (NDBR $\leq$ 0.2) or (RNIRB $\leq$ 4.5) or (NDVNIR $\leq$ 0.31) then: Pixel mask value = 0, Class 0, "no aerosol."
2) If (SLOPE2 $<$ 5000) and ($B16 < 1500$) then: Pixel mask value = 1, Class 1, "smoke endmembers."
3) If ($B16 > 3000$) then: Pixel mask value = 3, Class 3, "cloud endmembers."
4) Otherwise: Pixel mask value = 2, Class 2, "aerosol mixels."

The spectral indices are defined as follows, both in general wavelength regions, and in comparison to HS2 band numbers.

1) *Normalized difference vegetation index (NDVI):* NDVI = (NIR–RED)/(NIR + RED) = (B30–B16)/ (B30 + B16).
2) *Normalized difference red blue (NDBR):* NDBR = (RED–BLUE)/(RED + BLUE) = (B16–B6)/ (B16 + B6).
3) *Ratio NIR to blue (RNIRB):* RNIRB = NIR / BLUE = HB30/B6.
4) *Normalized difference violet NIR (NDVNIR):* NDVNIR = (VIOLET–NIR)/(VIOLET + NIR) = (B1–B29)/ (B1 + B29).
5) *Smoke-cloud spectral slope index 1 (SLOPE1):* SLOPE1 = (REDE–RED)/(0.744–0.648) = (B22–B16)/(0.744–0.648).
6) Smoke-cloud spectral slope index 2 (SLOPE2).

The SLOPE1 and SLOPE2 indices were developed here as they were found to differentiate key spectral features of clouds and smoke. Although the SLOPE1 index is not used explicitly in the classification of pixels values for smoke detection it was found to provide useful information postclassification in smoke-plume characteristics, particularly in identification of the near-source (fire) end of the plume.

B. VIB_SD Model

Our previously proposed fire smoke detection model, VIB_SD [14], is a lightweight AI model designed specifically for fire smoke detection using multispectral satellite imagery. VIB_SD integrates two key modules, respectively the inception-attention module and the inception-residual module, to facilitate residual learning and feature extraction at multiple scales, allowing an accurate detection of fire smoke in various scenarios from satellite imagery, even if the fire smoke is mixed with other types of aerosols, such as cloud or dust.

We chose VIB_SD as the onboard AI model for our emulation experiments for two primary reasons. First, VIB_SD is resource-efficient, featuring approximately 1.6 million parameters, in contrast to other state-of-the-art models that can have over 50 million parameters. This lightweight architecture makes it well-suited for onboard satellite detection. Second, VIB_SD has demonstrated high prediction accuracy in fire smoke detection tasks using multispectral satellite imagery. In a previous study [14], VIB_SD achieved an accuracy rate of 93.57% when using 368 Landsat-8 imagery with 256 $\times$ 256 resolution, classified into three categories: "Clear," "Other_aerosol," and "Smoke." Moreover, on synthetic HS2 hyperspectral imagery,

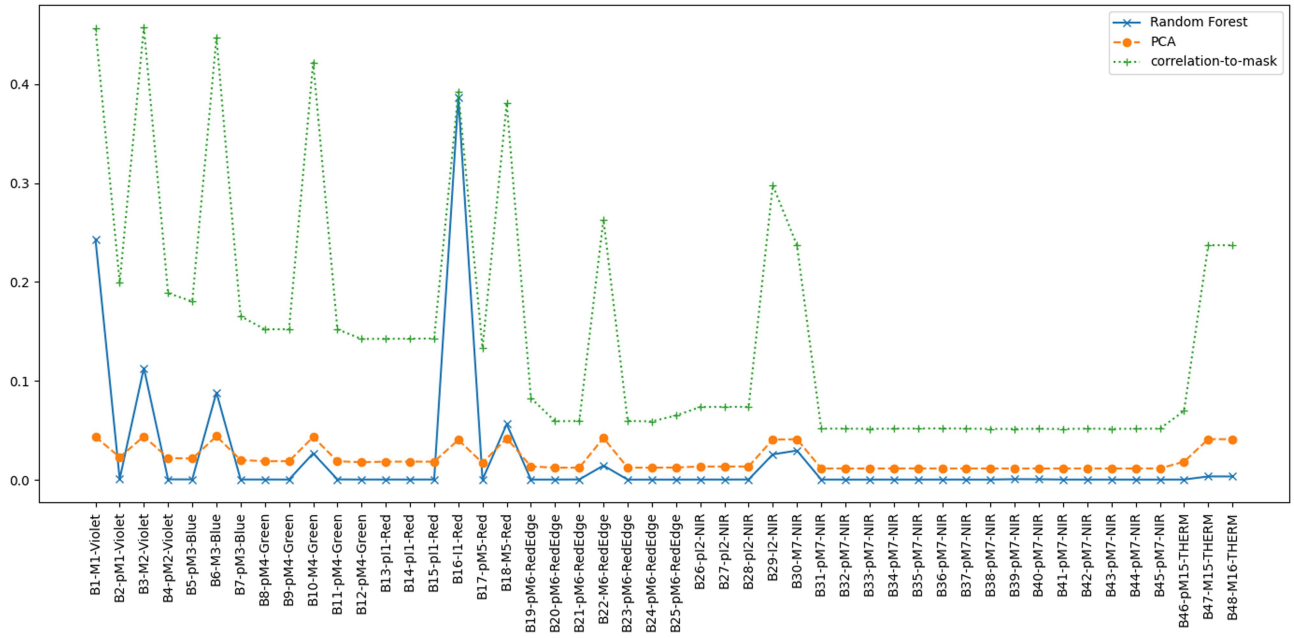

Fig. 2. Comparative visualization of band importance as determined by COR to the smoke mask, PCA, and RF methods in the context of band selection for AI onboard fire smoke detection. The three methods effectively identify the 11 original bands from VIIRS imagery but with different prioritization of bands by each method.

VIB_SD reached a prediction accuracy of 95.7% and had a FNR of only 2.3%. Readers can refer to Zhao et al.'s [14] work for more details about the VIB_SD model.

C. Band Selection

The HS2 sensor employed in the Kanyini satellite captures images comprising 45 visible and near-infrared (VNIR) bands and three thermal bands. However, when implementing AI onboard for fire smoke detection, the satellite does not have the capability to support a large AI model that utilizes input from all 48 bands. To address this limitation, we conduct on-ground based band selection to identify the bands that have the most significant impact on distinguishing between smoke and nonsmoke tiles. Only the identified bands are extracted from the tiles during AI inference onboard before being fed into the AI model.

To explore the effectiveness of different band selection techniques, we employed three commonly used techniques: Pearson correlation (COR), principal component analysis (PCA) [37], and random forest (RF) [38], [39]. Among these, PCA operates as an unsupervised method, utilizing all available data points irrespective of their labels. For COR, despite its typical categorization as an unsupervised technique, we adapted it for a supervised context by correlating features with the pixel classes, as described in Section III-A2. RF is a supervised method. Therefore, both COR and RF require labeled data for band selection. Band selection using COR and RF was conducted only using the training set.

The band importance derived from COR, PCA, and RF methods is depicted in Fig. 2. All three methods effectively identify the 11 original bands from the VIIRS imagery with higher importance for smoke detection, demonstrating the general effectiveness of these band selection techniques. A notable observation is the varying degrees of band prioritization across the methods. The RF method exhibits the largest variation in band importance, indicating a more distinct differentiation among bands. Conversely, the PCA method results in the least variation, suggesting a more uniform importance across the bands. This variation in the band importance show an impact on the performance of smoke detection (discussed in Section IV-B).

To evaluate the smoke detection performance of the VIB_SD model using various selected band sets, experiments were conducted with datasets comprising different band combinations. In addition to the three aforementioned band selection methods, two additional approaches were implemented: the ground truth (GT) method and the low band importance (LBI) method. The GT method involves selecting bands that are used to generating the pixel classes for labeling the tiles. Specifically, bands B16, B22, B29, B1, B6, and B30 were chosen for in the GT method, as detailed in Section III-A. This approach serves to benchmark the effectiveness of the band selection methods against an optimal scenario. The LBI method, on the other hand, includes bands consistently identified as of low importance by all three selection methods, providing insight into the potential impact of poor band selection on smoke detection performance. A summary of the bands selected by each of the five methods is provided in Table I. The smoke detection performance using different selected bands are analyzed and discussed in Section IV-B.

D. Emulation System

To assess the efficacy of onboard processes, we have constructed an emulation system that approximates the computational environment of the Kanyini satellite. On the hardware front, the system employs a Raspberry Pi (RPi) 4 Model B, featuring a Cortex-A72 (ARM v8) 64-bit SoC running at 1.5 GHz

TABLE I
TOP BANDS IDENTIFIED BY THE FIVE METHODS

	RF [1]	PCA [2]	COR [3]	GT [4]	LBI [5]
Top 1	B16-I1-Red	B6-M3-Blue	B3-M2-Violet	B16-I1-Red	B31-pM7-NIR
Top 2	B1-M1-Violet	B22-M6-RedEdge	B16-I1-Red	B22-M6-RedEdge	B32-pM7-NIR
Top 3	B18-M5-Red	B47-M15-THERM	B29-I2-NIR	B29-I2-NIR	B33-pM7-NIR
Top 4	B30-M7-NIR	B29-I2-NIR	B22-M6-RedEdge	B1-M1-Violet	B34-pM7-NIR
Top 5	B29-I2-NIR	B16-I1-Red		B6-M3-Blue	B35-pM7-NIR
Top 6				B30-M7-NIR	B36-pM7-NIR

[1] RF: Random forest; [2] PCA: Principle component analysis; [3] COR: Pearson correlation; [4] GT: Ground truth; [5] LBI: Low band importance.

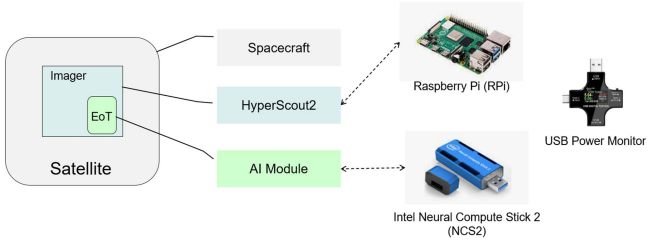

Fig. 3. Illustration of the emulation system hardware setup, featuring the Raspberry Pi as the HS2 sensor emulator and the Intel NCS2 as the AI module emulator.

with four cores, along with 4 GB of RAM. In addition, we utilize an Intel Neural Compute Stick 2 (NCS2), which is equipped with 16 SHAVEs and incorporates Intel Myriad X chip.

For software alignment, we configured the system to closely emulate the satellite specifications. The operation system of the Raspberry Pi is Raspberry Pi OS (Buster). We implemented a software pipeline that incorporates OpenVINO toolkit version 2021 [40]. This setup enables the evaluation of various performance metrics, such as execution time, memory consumption, and power usage across a broad spectrum of onboard tasks. Importantly, while the emulation system is tailored to the Kanyini satellite, it is architected to be flexible, thus allowing easy adaptation for other mission profiles.

1) Hardware of the Emulation System: A satellite computing environment typically consists of three levels: spacecraft, imager (or sensor), and AI module. In the Kanyini satellite, the imager employed is HS2, and the AI module utilizes the eyes of things (EOT) board developed by Ubotica.

The emulation system, designed to replicate the computing environment of a satellite, is composed of hardware components that simulate the imager and AI module. For the imager emulation, we use a Raspberry Pi to represent the HS2 sensor. The AI module is emulated using an Intel NCS2, simulating the functionality of the EOT board. Fig. 3 illustrates this hardware setup.

In configuring the emulation system, we selected hardware devices with specifications closely resembling or slightly superior to those anticipated in the satellite environment. The CPU frequency, number of CPU cores, RAM, and VPU cores of the emulation setup are carefully tuned to align with the expected Kanyini satellite environment via software configurations during the emulation experiments. It is important to note that discrepancies in the system might lead to marginally over- or underestimated performance results for the Kanyini satellite. However, these mismatches do not significantly impact

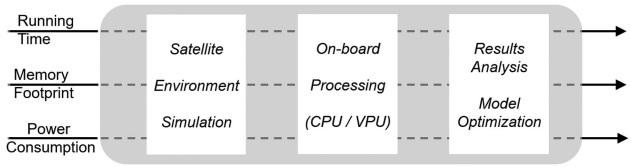

Fig. 4. Schematic representation of the emulation system software architecture, highlighting the four key modules: Performance measuring, resource constraint, onboard processing, and results analysis and model optimization.

the relative differences between results and will be addressed during a later phase of calibration.

2) Software of the Emulation System: The software architecture of the emulation system comprises four distinct modules, as illustrated in Fig. 4: 1) performance measuring; 2) resource constraint; 3) onboard processing; and 4) results analysis and model optimization. Together, these modules establish a complete pipeline for evaluating the performance of various onboard processing steps in the emulation environment.

The performance measuring module is designed to assess the running time, memory footprint, and power consumption of different onboard processes. Currently, power consumption measurements are conducted manually due to the lack of a software interface with the USB power monitor. Future research phases may include hardware upgrades to facilitate software-based power consumption measurement. The resource constraint module imposes hardware limitations on the emulation system to closely mimic the Kanyini satellite environment. These constraints include the CPU frequency, the number of CPU cores, and the RAM of RPi, as well as the number of VPU cores (SHAVES) of the NCS2. The onboard processing module executes various onboard tasks, encompassing both AI and non-AI processes. Finally, the results analysis and model optimization module collates and examines the test results, offering insights and recommendations for optimizing the AI model.

An exemplary emulation workflow begins by matching the hardware resources, and proceeds with the onboard processing sequence, which could include the following steps: 1) tiling and extracting specific bands from an image, 2) conducting AI inference, 3) creating a mask based on the AI inference results, 4) combining tiles with detected features, 5) compressing the merged image, and 6) dividing the compressed image into smaller segments in preparation for downlink transmission.

In order to facilitate onboard inference, the AI model needs to be converted into the intermediate representation (IR) format. This conversion is accomplished using the open visual inference and neural network optimization (OpenVINO) toolkit [41]. OpenVINO is a versatile open-source toolkit designed for optimizing deep learning models from various frameworks and deploying them across a wide range of Intel processors and hardware platforms, including the NCS2 used in our emulation system.

Fig. 5 illustrates the three-step process of preparing the AI model for onboard inference: model training, model converting, and model inference. The first two steps, model training and

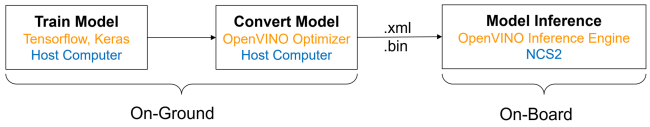

Fig. 5. Schematic representation of the AI model preparation process for onboard inference, detailing the steps of model training, model converting using OpenVINO optimizer, and model inference on the NCS2.

converting, are carried out on the ground. During model training, a model is trained and then saved on a host computer. In the model converting step, this trained model is transformed into the IR format using the OpenVINO optimizer. This process results in two files: an XML file (.xml), which represents the network topology, and a binary file (.bin) containing the network parameters. These files are then deployed onto the NCS2 for performing onboard inferences.

We have implemented the following onboard processes during the prelaunch phase.

1) *Compress imagery:* the emulation system supports the following compression methods: LZMA, LZW, PackBits, Deflate, PIXTIFF, LERC, Zstd, JPEG, JPEG2000, JPEGXL, and PNG.
2) Split imagery into small files for downlinking.
3) *Partition imagery into tiles:* the default tile dimension is 256 × 256, which is configurable.
4) Extract specified bands from imagery or tiles.
5) Merge specific tiles into the original imagery and mask the other tiles.
6) Onboard inference using VIB_SD model.

IV. Experimental Results and Discussions

The experiments in this study aim to validate the feasibility and benefits of onboard fire smoke detection, supported by experimental results. The experiments consist of two parts: 1) investigating the prediction accuracy of the VIB_SD model using different sets of selected bands, and 2) evaluating the onboard performance of various processes, including both non-AI and AI tasks.

A. Data Preparation

In the experiments, we use the simulated imagery described in Section III-A. The imagery is partitioned into tiles of a dimension of 256 × 256 pixels, consisting of 49 bands, which include 45 VNIR bands, three thermal bands, and a mask band. The mask band contains pixel-level labels, assigning each pixel to one of four classes: Zero for clear, one for smoke, two for mixed, and three for cloud.

As the VIB_SD model works to predict whether a tile contains smoke, the pixel-level label needs to be converted into a tile-level label. The tile-level labels consist of two classes: smoke and nonsmoke. If a tile contains any smoke pixels, it is categorized as smoke. Otherwise, it is identified as nonsmoke.

To generate datasets for our experiments, we sample 1600 tiles for each set of selected bands, as outlined in Section III-C. These 1600 tiles are evenly split between the two classes (smoke and nonsmoke), with 800 tiles in each. The sampling strategy aims to ensure a balanced distribution of tiles, covering a variety of scenarios and minimizing dataset bias. Consequently, a total of 21 distinct datasets are generated.

During the experiments, each dataset is randomly partitioned into a training set (comprising 60% of the tiles), a validation set (20%), and a test set (another 20%). Then, a VIB_SD model is trained on the training set and evaluated on the test set. This entire process is repeated ten times for each dataset, resulting in 210 trained models and their corresponding prediction results. From the ten prediction results obtained for each dataset, we report the average accuracy and FNR.

Accuracy is calculated using the formula: Accuracy = (TP + TN) / (TP + TN + FP + FN), where TP denotes true positives (correctly detected tiles with smoke), TN denotes true negatives (correctly identified tiles without smoke), FP denotes false positives (incorrectly detected tiles as having smoke when there is not), and FN denotes false negatives (incorrectly identified tiles as not having smoke when there is). This metric reflects the proportion of correctly identified tiles (both with smoke and without smoke) among the total number of tiles. For calculating the FNR, the formula is: FNR = FN / (FN + TP), which is a critical metric in this context as it indicates the rate of missed smoke detection events, essential for assessing the efficacy of the model.

B. Prediction Results

As discussed in Section III-C, we employed five band selection methods: COR, PCA, RF, GT, and LBI. In this section, we present and analyse the smoke detection performance using various selected band sets. Applying these five methods, we generated 21 distinct sets of bands by selecting 2–6 bands from each method (where applicable). These band sets were used to create corresponding datasets, named according to the band selection method and the number of bands selected, as indicated in Table I. For instance, COR4 refers to the dataset generated by selecting the top four bands from the COR method, i.e., B3-M2-Violet, B16-I1-Red, B29-I2-NIR, and B22-M6-RedEdge. Note that band selection was conducted only using the tiles in the training set.

The experimental results, summarized in Table II, are grouped by the number of bands selected, ranging from 2 to 6. In each group, the best results are highlighted in bold. Overall, selecting the top four bands demonstrated the highest performance, with COR and RF achieving the best accuracy and FNR, or close to it. A detailed analysis of each method reveals the following.

1) LBI consistently shows the poorest accuracy, indicating the necessity of band selection.
2) The GT method consistently performs well across all groups, as expected, confirming the importance of selecting the right bands for smoke detection. The highest performance from GT, an accuracy of 0.957 and FNR of 0.21 with four bands, was also achieved by COR and RF with certain band selections.
3) COR achieves the highest accuracy with three and four band selections, albeit with a slightly higher FNR than the optimal 0.21. However, its accuracy drops to 0.897 with

TABLE II
SMOKE DETECTION RESULTS USING VIB_SD WITH DIFFERENT SELECTED BANDS

BS-Method [1]	#Bands [2]	Accuracy [3]	FNR [4]
COR [5]	2	0.897 ± 0.013	0.068 ± 0.019
PCA [6]	2	**0.912** ± 0.016	**0.035** ± 0.016
RF [7]	2	0.890 ± 0.020	0.071 ± 0.019
GT [8]	2	**0.912** ± 0.016	0.038 ± 0.020
LBI [9]	2	0.550 ± 0.020	0.245 ± 0.091
COR [5]	3	**0.957** ± 0.010	**0.028** ± 0.014
PCA [6]	3	0.939 ± 0.012	0.030 ± 0.013
RF [7]	3	0.919 ± 0.024	0.042 ± 0.014
GT [8]	3	0.925 ± 0.016	0.031 ± 0.013
LBI [9]	3	0.555 ± 0.032	0.277 ± 0.085
COR [5]	4	**0.957** ± 0.008	0.023 ± 0.011
PCA [6]	4	0.943 ± 0.009	0.027 ± 0.006
RF [7]	4	**0.957** ± 0.010	**0.021** ± 0.010
GT [8]	4	**0.957** ± 0.009	**0.021** ± 0.011
LBI [9]	4	0.582 ± 0.025	0.248 ± 0.077
PCA [6]	5	**0.955** ± 0.011	**0.023** ± 0.014
RF [7]	5	0.953 ± 0.019	0.032 ± 0.015
GT [8]	5	0.954 ± 0.010	0.028 ± 0.010
LBI [9]	5	0.547 ± 0.023	0.218 ± 0.087
GT [8]	6	**0.950** ± 0.012	**0.023** ± 0.009
LBI [9]	6	0.560 ± 0.049	0.183 ± 0.115

[1] BS-Method: Band selection method; [2] #Bands: Number of bands; [3] Accuracy: Prediction accuracy; [4] FNR: False negative rate; [5] COR: Pearson correlation; [6] PCA: Principle component analysis; [7] RF: Random forest; [8] GT: Ground truth; [9] LBI: Low band importance.

only two bands. It is noted that COR is the one method attaining top accuracy with fewer than four bands.

4) While PCA does not reach the highest accuracy, it performs best (tied with GT) in the two-band group and maintains good results across all groups.
5) The performance of RF performance appears more sensitive to the number of selected bands, showing the best accuracy and FNR with four bands, but performing poorly (except when compared to LBI) with two bands.

In summary, these results suggest that while COR is generally a robust choice, PCA is a safer option across different scenarios. However, when using RF, the number of bands selected should be carefully considered.

C. Emulation Results

Emulation experiments were conducted using various parameters, including different numbers of pixels across the track for visual and near-infrared bands (selected from 3072 to 2560), different numbers of bands (selected from 45 to 48), and different numbers of tiles (selected from 5, 10, 20, and 40). The results of different onboard processes with these different configurations are presented Table III.

To evaluate the benefit with AI onboard, we use an example to compare the two scenarios: traditional smoke detection with AI on-ground and smoke detection with AI onboard. The input image has a resolution of 3072 × 1856 pixels and contains 45 bands. This image is simulated from a fire event near the Coorong in South Australia on December 31st, 2020. Approximately one-third of the image is covered by water, and about 5% shows fire smoke over land. Corresponding RGB and HSSDA figures are presented in Fig. 6.

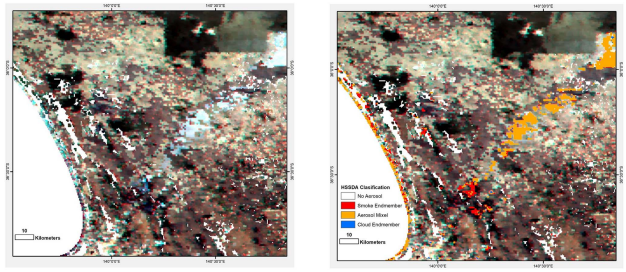

Fig. 6. HS2 simulated image over a fire event near the Coorong in South Australia on December 31st, 2020. Left: A "natural color" composite in red–green–blue wavelengths. Here the smoke plume appears in blueish gray tones near source, and white higher in the plume (visibly similar to cloud in other parts of the image). Burn scars are also hard to visibly separate from dark vegetation. Right: The HSSDA mask over the "natural color" composite. Here the two classes containing smoke are shown in red and orange. The plume is clearly detected, with the densest smoke near source classified as "smoke endmember." Smoke is not confused as being cloud. False detections occur near the shoreline.

Using the developed emulation system, we measure the running time, memory footprint, and power consumption associated with each step of the process workflow in the two scenarios. As illustrated in Fig. 7, this process of AI on-ground scenario begins with the compression of a captured image, having dimensions of 3072 × 1856 pixels and consisting of 45 bands (excluding thermal bands). The LZMA compression is then applied to reduce the original image size of 489 MB to a compressed file of approximately 388 MB. Our emulation reveals that this compression step takes about 656 s, consumes an average of 23 MB of memory with a peak at 530 MB, and has a power consumption of around 0.5 W.

Subsequently, the compressed image is divided into 778 smaller files, each 512 KB in size, to facilitate transmission (downlinking) to the ground station. The file splitting process is relatively fast, taking about 7 s, and requires an average memory of 22 MB with a peak at 83 MB. The power consumption during this step remains around 0.5 W. Once all these split files are successfully downlinked to the ground station, the smoke detection analysis is performed on the ground.

Fig. 8 showcases the simulated onboard smoke detection process using AI. In this setup, the process begins with tiling and band extraction from the input imagery, resulting in 84 tiles. Each tile, with dimensions of 256 × 256 pixels and comprising three selected bands, has a file size of about 0.38 MB. The tiling and band extraction step takes approximately 3.5 s, utilizes an average memory of 33 MB (peaking at 325 MB), and consumes around 0.5 W of power.

These 84 tiles then undergo AI model inference. This step of the process requires about 1.6 s, with an average memory usage of 29 MB, a peak memory usage of 54 MB, and a power consumption of 1.31 W. Assume that the AI model identifies ten tiles containing smoke. These detected tiles are then merged back into a single image, which retains the dimensions of the original

TABLE III
EMULATION RESULTS OF ONBOARD PROCESSES

Task	#ACT [1]	Size [2]	Output [3]	#Bands [4]	#Tiles [5]	Runtime [6]	Mem-A [7]	Mem-P [8]	Power [9]
compress	3072	489.38	38.15	45	5	462.964	23.73	530.00	0.56
compress	3072	489.38	60.53	45	10	655.716	22.95	529.57	0.44
compress	3072	489.38	105.12	45	20	691.346	23.28	529.73	0.42
compress	3072	489.38	194.31	45	40	739.537	25.82	532.20	0.52
compress	2560	407.82	35.53	45	5	538.534	23.11	447.81	0.48
compress	2560	407.82	57.86	45	10	551.944	23.58	448.30	0.52
compress	2560	407.82	102.54	45	20	575.366	23.39	448.10	0.48
compress	2560	407.82	146.86	45	30	602.159	24.26	449.03	0.49
compress	2560	407.82	191.30	45	40	562.194	26.39	451.25	0.45
compress	3072	489.38	388.58	45	45	833.857	22.14	529.06	0.47
compress	2560	407.82	323.82	45	45	690.647	22.06	447.54	0.46
compress	3072	489.38	149.60	45	30	527.425	23.34	531.55	0.51
merge_tiles	3072	489.38	489.38	45	5	6.287	29.64	519.41	0.53
merge_tiles	3072	489.38	489.38	45	10	6.175	26.04	519.24	0.49
merge_tiles	3072	489.38	489.38	45	20	6.235	26.56	518.98	0.44
merge_tiles	3072	489.38	489.38	45	30	6.161	29.23	518.99	0.46
merge_tiles	3072	489.38	489.38	45	40	6.107	29.37	519.13	0.45
merge_tiles	2560	407.82	407.82	45	5	5.291	29.29	437.40	0.47
merge_tiles	2560	407.82	407.82	45	10	5.269	29.52	437.73	0.47
merge_tiles	2560	407.82	407.82	45	20	5.196	29.36	437.56	0.43
merge_tiles	2560	407.82	407.82	45	30	5.141	29.30	437.50	0.44
merge_tiles	2560	407.82	407.82	45	40	5.029	29.49	437.70	0.41
smoke_mask	3072	489.38	10.88	45	10	2.734	29.19	527.05	0.51
smoke_mask	2560	407.82	9.06	45	10	2.345	28.97	444.15	0.48
split	3072	38.15	0.50	45	5	0.682	22.50	61.07	0.44
split	3072	60.53	0.50	45	10	1.09	22.40	83.29	0.52
split	3072	105.12	0.50	45	20	1.886	22.55	128.04	0.48
split	3072	194.31	0.50	45	40	3.509	22.60	217.29	0.42
split	2560	35.53	0.50	45	5	0.661	22.45	58.34	0.45
split	2560	57.86	0.50	45	10	1.055	22.78	81.09	0.41
split	2560	102.54	0.50	45	20	1.835	22.55	125.47	0.42
split	2560	146.86	0.50	45	30	2.619	22.69	169.94	0.5
split	2560	191.30	0.50	45	40	3.446	22.54	214.13	0.46
split	3072	388.58	0.50	45	45	6.713	22.23	411.38	0.52
split	2560	323.82	0.50	45	45	5.69	22.69	347.08	0.48
split	3072	149.60	0.50	45	30	2.58	22.47	172.62	0.46
tiling&extract_bands	3072	489.38	0.38	3	84	3.478	33.31	523.25	0.47
tiling&extract_bands	3072	489.38	0.51	4	84	3.633	32.89	523.84	0.44
tiling&extract_bands	3072	489.38	0.63	5	84	3.635	32.74	523.95	0.46
tiling&extract_bands	2560	407.82	0.38	3	70	2.889	33.24	441.88	0.55
tiling&extract_bands	2560	407.82	0.51	4	70	3.059	32.67	441.90	0.46
tiling&extract_bands	2560	407.82	0.63	5	70	3.07	32.74	442.30	0.47
VIB_SD_inference		6.29				1.582	29.51	54.52	1.31

[1] #ACT: the number of pixels of VNIR bands on across the track; [2] Size: the imagery file size in MiB; [3] Output: the output file size in MiB; [4] #Bands: the number of bands; [5] #Tiles: the number of tiles; [6] Runtime: running time in seconds; [7] Mem-A: the average memory in MiB; [8] Mem-P: the peak memory in MiB; [9] Power: the power consumption in Watt.

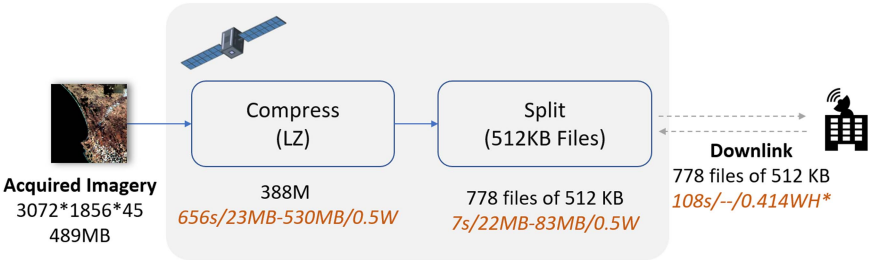

Fig. 7. Simulation results showcasing the workflow of traditional smoke detection (AI on-ground), including the compression and downlinking steps prior to ground-based smoke detection analysis.

imagery, band selection, and georeferencing information. The merging process consumes 6.2 s, an average memory of 26 MB, peaks at 519 MB, and requires approximately 0.5 W of power.

Following the merging, the next step is the compression of this merged image from sized at 489–61 MB. The compressed file is then segmented into 122 smaller files, each of 512 KiB, ready for downlinking to the ground station. For data downlinking, assuming the downlinking data rate is 28.7 Mbps, and the downlink power is 13.8 W (derived from the Φ-Sat-1 mission [13]), the energy consumption for data downlinking in the example of the AI onboard scenario is 0.065 WH, while the AI on-ground scenario requires 0.414 WH.

Table IV compares the resource consumption between the two scenarios, AI onboard and AI on-ground. The traditional

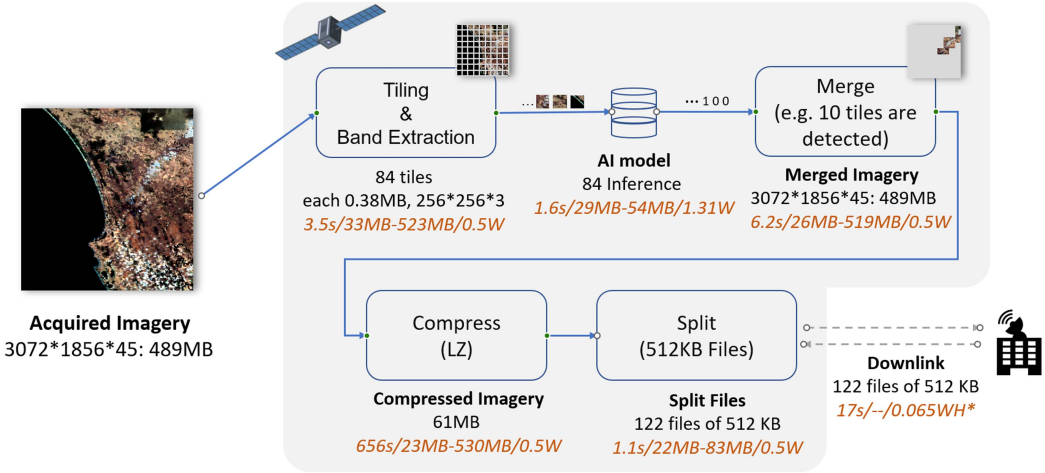

Fig. 8. Simulation of the onboard smoke detection process using AI, detailing the steps of tiling, band extraction, AI model inference, image merging, compression, and file segmentation for downlinking, along with their respective time, memory, and power requirements.

TABLE IV
COMPARISON OF RESOURCE CONSUMPTION IN AI ONBOARD VERSUS AI ON-GROUND SCENARIOS

Step	AI on-ground [1]	AI onboard [2]
Tiling and band extraction	–	3.5 s / 33 –523 MB / 0.5 W
AI inference	–	1.6 s / 29 –54 MB / 1.31 W
Merge	–	6.2 s / 26 –519 MB / 0.5 W
Compress	656 s / 23 MB–530 MB / 0.5 W	656 s / 23 –530 MB / 0.5 W
Split	7 s / 22 MB–83 MB / 0.5 W	1.1 s / 22 –83 MB / 0.5 W
Processing [3]	663 s, 0.092WH	668 s, 0.093WH
Downlink [4]	388 MB, 108 s, 0.414WH	61 MB, 17 s, 0.065WH

[1] in format of running time (s)/average memory (MB)-peak memory (MB)/power consumption (W);
[2] in format of running time (s)/average memory (MB)-peak memory (MB)/power consumption (W), assuming that 10% of the original imagery contains smoke;
[3] in format of running time (s), total energy consumption in Watt-Hour (WH);
[4] in format of downlink volume (MB), transferring time(s), energy consumption in Watt-Hour (WH). Calculation assumes downlink data rate is 28.7 Mbps and downlink power consumption is 13.8 W.

scenario wit AI on-ground, requires downlinking the entire compressed image, sized at 388 MB. This process incurs significant resource usage for data downlinking, notably a transfer time of 108 s and an energy expenditure of 0.414 WH. In contrast, the AI onboard scenario demonstrates a marked efficiency improvement, which reduces the downlink data volume to 61 MB with downlink time to 17 s and energy consumption to 0.065 WH. Notably, this reduced data volume is particularly advantageous given the constraints on downlinking capacity during each satellite pass over a ground station, potentially avoiding delays inherent in waiting for subsequent passes for data transmission.

Although the introduction of AI onboard introduces additional steps, such as tiling, band extraction, and AI inference, these steps are executed relatively fast (3.5 and 1.6 s, respectively) and with small power consumption. The most time-consuming step, compression, remains constant in both scenarios, ensuring that the overall processing time and energy consumption are only marginally increased (from 663 to 668 s, from 0.092 to 0.093 WH) when AI is onboard. The slight increase in processing time is outweighed by the substantial benefits in downlink efficiency.

In summary, the implementation of AI onboard for smoke detection exhibits clear advantages over traditional methods. These benefits primarily manifest in reduced downlink requirements, leading to significant savings in processing time and energy.

D. Discussion

This discussion delves into the deeper implications, limitations, and practical applicability of our findings. The VIB_SD model has demonstrated high prediction accuracy and a low FNR in simulated datasets. However, its true effectiveness will be more accurately evaluated when applied to actual satellite data from the Kanyini mission.

Real-world conditions present several challenges that could potentially affect the model performance. These include atmospheric variability, cloud cover, and sensor noise. Implementing atmospheric correction onboard satellites could significantly increase timing and energy consumption. Conversely, omitting this step might impair the model accuracy. Although the onboard data handling system of the Kanyini satellite being equipped with coarse georeferencing and atmospheric correction capabilities, there remains a potential for discrepancies in accuracy and energy efficiency. Future research should focus on validating the model performance under varying and unpredictable environmental conditions.

In addition, while our emulation results are indicative, they are subject to variables, such as hardware and software discrepancies, and coding practices. Notably, the difference in VPU chips could cause variations in AI model performance. This necessitates a meticulous calibration of the emulation system to better align with the satellite computing environment. Gaining access to the satellite or its engineering model will be crucial for this calibration, ensuring our results more accurately reflect the operational realities of the Kanyini satellite.

An important finding of our study is the significant reduction in downlink time and energy consumption achieved by integrating AI onboard. This improvement is crucial for enhancing data transmission efficiency in satellite operations. However, it is accompanied by an increased demand for onboard processing time due to AI inference. Balancing downlink efficiency with the additional onboard processing requirements is important, especially for missions with limited power resources or downlink opportunities.

Furthermore, our study highlights another significant benefit of onboard AI: the early detection of fire smoke. Traditional methods typically yield smoke detection results only after the imagery is downlinked and processed on the ground, which can delay response times. In contrast, AI onboard enables immediate fire smoke detection directly on the satellite, a capability that is particularly advantageous in satellite constellations. This enables rapid transmission of detection results between satellites, facilitating early warning systems for wildfires and other urgent environmental monitoring tasks. This proactive detection capability enhances the responsiveness of satellite-based monitoring systems and contributes to timely decision-making and intervention in emergencies.

In conclusion, our research presents valuable insights into the potential of AI-enhanced fire smoke detection for satellite applications. However, it also highlights the imperative of rigorous real-world testing and calibration to fully realize and optimize this technology for practical use in satellite missions.

V. Conclusion and Future Work

In conclusion, this research project has provided a solution for energy-efficient AI-based onboard processing of hyperspectral imagery for early fire smoke detection, especially for the prelaunch stage. The deployment of the VIB_SD model, operating within the constraints of the HS2 sensor on the Kanyini satellite, has demonstrated promising results in terms of high prediction accuracy and low FNR. The simulation of a comprehensive training dataset and the implementation of an emulation system have further confirmed the feasibility and benefits of onboard processing. The significant reduction in data downlink volume and energy consumption, coupled with the increased speed of fire smoke detection achieved through AI onboard, highlight the practical advantages of onboard AI-based fire smoke detection. The findings of this research not only contribute to advancing satellite-based early fire smoke detection capabilities but also hold promise for enhancing wildfire monitoring and response efforts.

Future work in the postlaunch phase involves calibrating the emulation system to match the actual environment of Kanyini. In addition, there is a need for model implementation and stepwise model updates using real Kanyini imagery training data after the launch. Exploring alternative AI models, such as segmentation models, can also be considered. Finally, conducting a comparison between AI-based and deterministic (spectral index-based smoke detection) onboard methods would provide valuable insights.

Moreover, onboard change detection techniques utilizing time series imagery have demonstrated promising outcomes [42], [43], [44]. However, in our current study, the implementation of such techniques was constrained by the limited power and storage capacities of the satellite platform. Future research will explore these advanced change detection methodologies, such as [45] and [46], particularly for fire smoke detection.

Code and simulation results are available at[1] (access can be provided on request). More details can be found in Stefan et al.'s [33] work.

References

[1] F. Tedim et al., "Defining extreme wildfire events: Difficulties, challenges, and impacts," *Fire*, vol. 1, no. 1, pp. 1–28, 2018.

[2] N. Borchers Arriagada, A. J. Palmer, D. M. Bowman, G. G. Morgan, B. B. Jalaludin, and F. H. Johnston, "Unprecedented smoke-related health burden associated with the 2019–20 bushfires in eastern Australia," *Med. J. Aust.*, vol. 213, no. 6, pp. 282–283, Sep. 2020.

[3] Y.-S. Chung and H. Le, "Detection of forest-fire smoke plumes by satellite imagery," *Atmospheric Environ. (1967)*, vol. 18, no. 10, pp. 2143–2151, 1984.

[4] I. Nagatani and J.-I. Kudoh, "A false-color composite method for wildfire smoke plume identification using modis data," *J. Remote Sens. Soc. Jpn.*, vol. 33, no. 1, pp. 38–47, 2013.

[5] S. Chen, Y. Cao, X. Feng, and X. Lu, "Global2salient: Self-adaptive feature aggregation for remote sensing smoke detection," *Neurocomputing*, vol. 466, pp. 202–220, 2021.

[6] R. Ba, C. Chen, J. Yuan, W. Song, and S. Lo, "SmokeNet: Satellite smoke scene detection using convolutional neural network with spatial and channel-wise attention," *Remote Sens.*, vol. 11, no. 14, 2019, Art. no. 1702.

[7] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[8] B. Zhang et al., "Progress and challenges in intelligent remote sensing satellite systems," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 1814–1822, 2022.

[9] G. Giuffrida et al., "CloudScout: A deep neural network for on-board cloud detection on hyperspectral images," *Remote Sens.*, vol. 12, no. 14, 2020, Art. no. 2205.

[10] G. Mateo-Garcia et al., "Towards global flood mapping onboard low cost satellites with machine learning," *Sci. Rep.*, vol. 11, no. 1, 2021, Art. no. 7249.

[11] S. A. S. I. Centre, "Sa space services mission: SASAT1," 2023. [Online]. Available: https://www.saspacemission.com.au/

[12] 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 SmallSats Remote Sens. III*, vol. 11131, 2019, pp. 88–96.

[13] G. Giuffrida et al., "The ϕ-Sat-1 mission: The first on-board deep neural network demonstrator for satellite Earth observation," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5517414.

[14] L. Zhao, J. Liu, S. Peters, J. Li, S. Oliver, and N. Mueller, "Investigating the impact of using IR bands on early fire smoke detection from landsat imagery with a lightweight CNN model," *Remote Sens.*, vol. 14, no. 13, 2022, Art. no. 3047.

[15] C. Gómez, J. C. White, and M. A. Wulder, "Optical remotely sensed time series data for land cover classification: A review," *ISPRS J. Photogrammetry Remote Sens.*, vol. 116, pp. 55–72, 2016.

[16] Y. Cai et al., "A high-performance and in-season classification system of field-level crop types using time-series landsat data and a machine learning approach," *Remote Sens. Environ.*, vol. 210, pp. 35–47, 2018.

[17] M. Gazzea, M. Pacevicius, D. O. Dammann, A. Sapronova, T. M. Lunde, and R. Arghandeh, "Automated power lines vegetation monitoring using high-resolution satellite imagery," *IEEE Trans. Power Del.*, vol. 37, no. 1, pp. 308–316, Feb. 2022.

[18] O. Mutanga, T. Dube, and F. Ahmed, "Progress in remote sensing: Vegetation monitoring in South Africa," *South Afr. Geographical J.*, vol. 98, no. 3, pp. 461–471, 2016.

[1] [Online]. Available https://github.com/ShaLu-ML/SS-P2.38

[19] L. G. Olmanson, P. L. Brezonik, and M. E. Bauer, "Remote sensing for regional lake water quality assessment: Capabilities and limitations of current and upcoming satellite systems," in *Adv. Watershed Sci. Assessment*, pp. 111–140, 2015.

[20] G. J. Schumann, G. R. Brakenridge, A. J. Kettner, R. Kashif, and E. Niebuhr, "Assisting flood disaster response with Earth observation data and products: A critical assessment," *Remote Sens.*, vol. 10, no. 8, 2018, Art. no. 1230.

[21] S. Zhao et al., "The role of satellite remote sensing in mitigating and adapting to global climate change," *Sci. Total Environ.*, 2023, Art. no. 166820.

[22] J. Yang et al., "The role of satellite remote sensing in climate change studies," *Nat. Climate Change*, vol. 3, no. 10, pp. 875–883, 2013.

[23] M. Ghiglione and V. Serra, "Opportunities and challenges of ai on satellite processing units," in *Proc. 19th ACM Int. Conf. Comput. Front.*, 2022, pp. 221–224.

[24] C. Salazar et al., "Cloud detection autonomous system based on machine learning and cots components on-board small satellites," *Remote Sens.*, vol. 14, no. 21, 2022, Art. no. 5597.

[25] 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 Sens.*, vol. 13, no. 17, 2021, Art. no. 3479.

[26] G. Mateo-Garcia et al., "In-orbit demonstration of a re-trainable machine learning payload for processing optical imagery," *Sci. Rep.*, vol. 13, no. 1, 2023, Art. no. 10391.

[27] J. F. Munoz-Martin et al., "In-orbit validation of the FMPL-2 instrument–The GNSS-R and l-band microwave radiometer payload of the FSSCat mission," *Remote Sens.*, vol. 13, no. 1, p. 121, 2020.

[28] A. Camps et al., "FSSCat mission description and first scientific results of the FMPL-2 onboard 3CAT-5/A," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2021, pp. 1291–1294.

[29] A. M. Wijata et al., "Taking artificial intelligence into space through objective selection of hyperspectral Earth observation applications: To bring the "brain" close to the "eyes" of satellite missions," *IEEE Geosci. Remote Sens. Mag.*, vol. 11, no. 2, pp. 10–39, Jun. 2023.

[30] M. Esposito and A. Z. Marchi, "In-orbit demonstration of the first hyperspectral imager for nanosatellites," in *Proc. Int. Conf. Space Optics–ICSO 2018*, 2019, vol. 11180, pp. 760–770.

[31] K. Thangavel et al., "Autonomous satellite wildfire detection using hyperspectral imagery and neural networks: A case study on Australian wildfire," *Remote Sens.*, vol. 15, no. 3, p. 720, 2023.

[32] 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 *Proc. IEEE Int. Conf. Metrol. Extended Reality, Artif. Intell. Neural Eng. (MetroXRAINE)*, 2022, pp. 725–730.

[33] P. Stefan et al., "Technical report: Energy-efficient onboard AI for early fire-smoke detection," SmartSat Cooperative Centre, Tech. Rep. 11, 2023. https://smartsatcrc.lbcdn.io/uploads/Tech-Report-11-Energy-efficient-onboard-AI-for-early-fire-smoke-detection-2.pdf

[34] "Appeears - application for extracting and exploring analysis ready samples," LP DAAC (Land Processes Distributed Active Archive Center). Accessed: May 2022. [Online]. Available: https://lpdaac.usgs.gov/tools/appeears/

[35] M. H. Syed, K. Upreti, M. S. Nasir, M. S. Alam, and A. Kumar Sharma, A. "Addressing image and Poisson noise deconvolution problem using deep learning approaches," *Comput. Intell.*, vol. 39, no. 4, pp. 577–591, 2023.

[36] X. Lu, X. Zhang, F. Li, M. A. Cochrane, and P. Ciren, "Detection of fire smoke plumes based on aerosol scattering using VIIRS data over global fire-prone regions," *Remote Sens.*, vol. 13, no. 2, p. 196, 2021.

[37] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," *Chemometrics Intell. Lab. Syst.*, vol. 2, no. 1–3, pp. 37–52, 1987.

[38] J. L. Speiser, M. E. Miller, J. Tooze, and E. Ip, "A comparison of random forest variable selection methods for classification prediction modeling," *Expert Syst. with Appl.*, vol. 134, pp. 93–101, 2019.

[39] J. C.-W. Chan and D. Paelinckx, "Evaluation of random forest and adaboost tree-based ensemble classification and spectral band selection for ecotope mapping using airborne hyperspectral imagery," *Remote Sens. Environ.*, vol. 112, no. 6, pp. 2999–3011, 2008.

[40] I. Corporation, "Openvino toolkit." Accessed: May 2022. [Online]. Available: https://software.intel.com/en-us/openvino-toolkit

[41] Intel Corporation, "Openvino documentation - get started." Accessed: May 2022. [Online]. Available: https://docs.openvino.ai/2023.0/get_started.html

[42] Z. Lv, H. Huang, W. Sun, M. Jia, J. A. Benediktsson, and F. Chen, "Iterative training sample augmentation for enhancing land cover change detection performance with deep learning neural network," *IEEE Trans. Neural Netw. Learn. Syst.*, 2023.

[43] M. Chang, X. Meng, W. Sun, G. Yang, and J. Peng, "Collaborative coupled hyperspectral unmixing based subpixel change detection for analyzing coastal wetlands," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 8208–8224, 2021.

[44] K. Wu, Y. Zhong, X. Wang, and W. Sun, "A novel approach to subpixel land-cover change detection based on a supervised back-propagation neural network for remotely sensed images with different resolutions," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 10, pp. 1750–1754, Oct. 2017.

[45] K. He et al., "A dual global–local attention network for hyperspectral band selection," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, 2022, Art. no. 5527613.

[46] W. Sun, K. Ren, X. Meng, G. Yang, J. Peng, and J. Li, "Unsupervised 3-D tensor subspace decomposition network for spatial-temporal-spectral fusion of hyperspectral and multispectral images," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5528917.