POLITECNICO DI TORINO
Repository ISTITUZIONALE

Efficient Onboard Multitask AI Architecture Based on Self-Supervised Learning

Original
Efficient Onboard Multitask AI Architecture Based on Self-Supervised Learning / Inzerillo, G., Valsesia, D., Magli, E.. - In: IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING. - ISSN 1939-1404. - 18:(2025), pp. 828-838. [10.1109/jstars.2024.3502776]

Availability:
This version is available at: 11583/2995171 since: 2024-12-11T07:50:59Z

Publisher:
IEEE

Published
DOI:10.1109/jstars.2024.3502776

Terms of use:

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

Publisher copyright

(Article begins on next page)

20 November 2026

Efficient Onboard Multitask AI Architecture Based on Self-Supervised Learning

Gabriele Inzerillo, Diego Valsesia, *Member, IEEE*, and Enrico Magli, *Fellow, IEEE*

***Abstract*—There is growing interest toward the use of artificial intelligence (AI) directly onboard satellites for quick analysis and rapid response to critical events such as natural disasters. This article presents a blueprint to the mission designer for the development of a modular and efficient deep learning payload to address multiple onboard inference tasks. In particular, we design a self-supervised lightweight backbone that provides features to efficient task-specific heads. The latter can be developed independently and with reduced data labeling requirements thanks to the frozen backbone. Experiments on three sample tasks of cloud segmentation, flood detection, and marine debris classification on a 7-W embedded system show competitive results with inference quality close to high-complexity state-of-the-art models and high throughput in excess of 8 Megapixel/s.**

***Index Terms*—Multitask learning, onboard AI, self-supervised learning (SSL).**

I. Introduction

IN CONVENTIONAL satellite imaging systems, the satellite's task is to capture data, typically images, and transmit them to the ground segment for processing into various levels of products to be delivered to the final users. This transmission and processing chain can result in significant delays, in the order of days, to the availability of imagery to end users. This is especially undesirable in time-sensitive problems, such as natural disasters, where it is critical to obtain the data as soon as possible.

An emerging paradigm [1] is to move part of the processing directly onboard the satellite so that it can detect potentially critical situations in real time and trigger early warnings whose quick transmission to the ground segment is prioritized. This process requires at least partial onboard formation of image products, followed by image analysis; a full onboard pipeline of optical and synthetic aperture radar (SAR) image formation, analysis, and alert generation has been demonstrated in [2]. Achieving this goal requires facing a challenging tradeoff between the quality of the detection, its latency, and the computational constraints of onboard platforms dictated by the strict power budgets of satellites. Moreover, multispectral images are capable of detecting several phenomena of interest, such as floods, fires, clouds, marine debris, and many more, leading to the capability of addressing multiple tasks at the same time. Indeed, onboard multitask inference would not only allow to monitor multiple critical phenomena at the same time but also synergistically improve the entire platform. For example, segmentation of clouds could be used to optimize the onboard compression algorithm by lowering the data rate for cloud-covered areas.

Deep learning is key to achieving state-of-the-art performance for the detection tasks we aim at solving. However, designing a mission with its onboard use with real-time performance for multiple tasks is far from trivial and requires careful study of multiple issues. In this article, we study how a potential mission could define an artificial intelligence (AI) computational payload providing low-latency responses to multiple tasks with efficient use of resources and flexible design. In particular, we envision the use of a neural network composed of a lightweight backbone to extract features from multispectral input images at multiple spatial resolutions, including relatively fine-grained ones. This feature extractor is trained in a self-supervised manner to exploit large collections of unlabeled imagery by the mission operations center, and the model is made available to entities (e.g., third-party contractors), or it is made publicly available. The features are then used by lightweight neural network heads, working in parallel, each specialized for one image analysis task. These heads can be designed independently by third-party contractors, with domain knowledge of the tasks. The third-party contractors will be required to use the backbone without the ability to change its weights so that multiple heads can share the features for their respective tasks. This approach also conveniently limits the data requirements for the third parties who have to develop the task-specific heads, since they can leverage the backbone features and only need a small amount of labels to train the small heads. Once deployed onboard, the architecture can solve as many task as the number of heads in parallel, but conveniently sharing features to significantly reduce the computational requirements. Finally, the modular approach allows in-flight updating of the backbone and heads, or even addition of new tasks.

To summarize, this article presents the following key novel contributions.

1) We study how to design the onboard AI system for an Earth observation mission required to address multiple tasks, analyzing the entire framework needed to accomplish this goal, the technical details of the individual components, and presenting novel methodologies.
2) We show that a desirable design pretrains a backbone neural network with a self-supervised strategy, and in

Received 7 March 2024; revised 29 August 2024 and 2 October 2024; accepted 15 November 2024. Date of publication 20 November 2024; date of current version 6 December 2024. This work was supported in part by the Italian Space Agency under Agenzia Spaziale Italiana (ASI) contract 2023-23-U.0 within the programme on "Analisi dati e immagini," and in part by the CINECA award under the ISCRA initiative. *(Corresponding author: Diego Valsesia.)*

The authors are with the Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Torino, Italy (e-mail: diego.valsesia@polito.it).

Digital Object Identifier 10.1109/JSTARS.2024.3502776

© 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/

contrast with classical self-supervised learning (SSL) literature, keeps it frozen to allow processing multiple tasks in parallel.
3) We propose a novel SSL method that can extract features with both fine and coarse spatial resolution, outperforming existing SSL methods in remote sensing.
4) We evaluate the effects of quantization on SSL pretrained models, a topic rarely explored in the SSL literature.
5) We demonstrate a lightweight and modular design that provides inference accuracy close to that of high-complexity state-of-the-art models while achieving higher throughput on three tasks for onboard computing (clouds segmentation, floods segmentation, and marine debris classification)
6) We demonstrate that the proposed design has an excellent tradeoff between quality, throughput, and power consumption on a low-power 7-W embedded system.

II. Background

A. SSL

SSL has emerged in the last years as a powerful paradigm in deep learning, aiming at learning good representations that capture intrinsic data features without relying on human-labeled annotations. This is critical in remote sensing due to scarcity of labeled data and the abundance of unlabeled imagery. Contrastive learning is currently one of the most successful SSL techniques where informative representations emerge from minimizing the distance between the feature-space embedding of the same image subjected to two distinct random augmentations (positive pair) while maximizing the distance between representations derived from distinct images (negative pairs). Works such as SimCLR [3] highlight the efficacy of contrastive methods in learning resilient and generalizable representations, although they are not without flaws. In fact, they often need a large batch size to work properly and also the handling of negative pairs needs to be carefully managed. The need of a large batch size was partially solved by Momentum Contrast (MoCo) [4], using a momentum moving average encoder. Newer methods such as bootstrap your own latent (BYOL) [5] also address the problem of creating truly negative pairs by relying only on positive ones. Furthermore, He et al. [6] observed that image-level learning does not always provide good representations for subpixel level tasks such as semantic segmentation and object detection limiting the effectiveness of SSL only to image-level tasks such as classification. For this reason, research has started investigating "dense SSL" techniques [7], [8], [9] capable of learning more fine-grained features.

In the context of remote sensing, some works [10], [11], [12], [13] have explored pretext tasks and ways of framing contrastive learning that lead to SSL features that are more suitable for the remote sensing detection tasks. It is also worth noting that a typical framework for most works is to use SSL as a pretraining technique, followed by supervised finetuning of the entire model, including application head. This typically results in better accuracy than what would be obtained by keeping the backbone frozen to the SSL-trained weights. However, in this work, we will not follow this finetuning approach as it poses undesirable restrictions in the mission design, such as the inability to develop heads independently.

B. Efficient Inference With Neural Networks

The pursuit of efficient inference in deep learning has spurred numerous innovations in the realm of lightweight networks and quantization techniques. Lightweight architectures such as MobileNet [14], ShuffleNet [15], and EfficientNet [16] have aimed at reducing computational overhead while preserving accuracy. These networks employ strategies such as depthwise separable convolutions, channel shuffling, and compound scaling to achieve a balance between model size and accuracy. On the other hand, quantization techniques, such as posttraining quantization [17] and quantization-aware training [18], aim to reduce model size and increase inference speed by representing weights and activations using lower bit precision. In addition, methods such as knowledge distillation [19] and neural architecture search [20] have also been used in crafting efficient networks, either by transferring knowledge from larger models to smaller ones or by automating the design process to discover architectures optimized for fast inference.

C. Onboard AI-Based Processing

In recent years, the advancement of neural networks and AI has extended to onboard satellite processing systems and edge devices in general, enabling real-time data analysis directly in space. This advancement is particularly significant in the context of remote sensing, where the ability to process large volumes of onboard imagery can dramatically reduce latency, optimize bandwidth utilization, and enable more responsive and autonomous satellite operations. However, onboard AI is faced with several challenges in the design of lightweight and power-efficient systems. Several studies [21] have started demonstrated the feasibility and effectiveness of AI-based onboard processing for remote sensing tasks. Yao et al. [22] was one of the first works to address the challenge of running deep learning models directly onboard satellites, proposing a simple framework for ship detection on small satellites. Notably, Giuffrida et al. [1] demonstrated the integration of AI for onboard data processing in real Earth observation missions (Φ-Sat-1 by the European Space Agency), showcasing the feasibility of running deep convolutional neural networks (CNNs) on the Intel Movidius Myriad 2 hardware accelerator for real-time cloud detection on hyperspectral images. Ziaja et al. [23] proposed an extensive benchmark of various deep learning models on edge devices for onboard space applications. Růžička et al. [24] introduced a lightweight model for change detection onboard satellites based on variational autoencoders.

In this article, as a demonstration of the performance of the proposed design in a low-power setting, we test using an Nvidia Jetson Orin Nano, a commercial-off-the-shelf (COTS) hardware platform. This should be considered as a low-power demonstrator, and not necessarily representative of a real flight implementation. Indeed, we recognize that different space missions may choose different approaches to the integration of deep

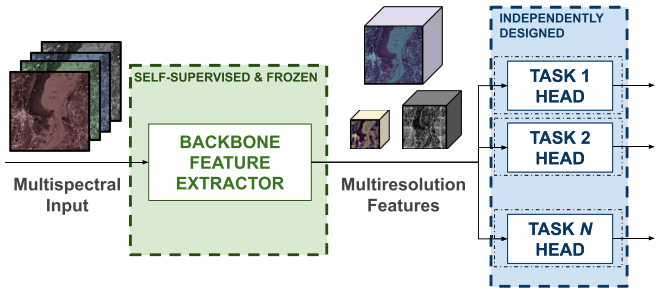


Fig. 1. High-level design for modular multitask neural network design. A lightweight backbone is trained with SSL, and then, frozen to generate universal standard multiresolution features. Application-specific heads can be independently developed to exploit such features for inference tasks.

learning in the onboard computing platforms depending on specific mission characteristics. For instance, they may rely solely on field programmable gate array (FPGAs), or a combination of FPGAs with graphic processing unit (GPUs)/central processing unit (CPUs) or even just a COTS system-on-chip.

III. Method

In this section, we introduce a novel, modular and lightweight multitask architecture tailored for usage onboard satellites for low-latency inference. We also go beyond the mere architecture design by presenting ideas that serve as a blueprint for a mission planning, which in turn affect decisions about the neural network development.

A. Mission Vision

The approach toward the design of neural network components described in later sections stems from ideas about the specific goals and planning requirements of a hypothetical mission. We envision a Sentinel2-like multispectral imager with additional capabilities provided by onboard AI. In particular, the first novel capability would be AI-assisted onboard compression. It is known that image compression methods, including existing standards for hyper- and multispectral images, can be aided by cloud detection [25] to provide pixel-level maps of regions where compression quality can be lowered to significantly save data rate. Cloud segmentation is, therefore, a desirable task to be included for any onboard AI capability. Furthermore, the second capability of interest is the generation of alerts to be delivered to the ground segment with low latency when specific phenomena, such as natural disasters, are detected. For this capability, it is desirable to produce both pixel-level segmentation maps (e.g., to detect the extent of flooded areas) as well as whole-image classification labels (e.g., to detect debris presence and its type, or the presence of active fires). These requirements clearly outline the need to have features with fine spatial granularity so that the segmentation tasks can be solved effectively.

Concerning mission planning, a modular approach is required so that multiple parties can cooperate in the design of the AI module and its possible update. In particular, Fig. 1 highlights multiple modules to be developed independently. A backbone serves as a universal feature extractor. This is developed independently of the specific tasks to be solved, except for the requirement of providing features with fine-grained spatial resolution. The features extracted by the backbone for a specific input are then used by task-specific heads that are comparatively smaller neural networks. These can be assigned to multiple third-party domain experts for their development. However, in order to guarantee reusability of the features for all tasks, such third parties are not allowed to fine-tune the backbone.

Finally, the entire neural model must be lightweight so that it can fit the limited memory of embedded systems and provide a high enough throughput. Targets for throughput depend on the specific mission requirements in terms of coverage and latency. However, as a rough idea, we can consider as generally adequate a throughput in the same order of magnitude of that of the image compression subsystem that is designed to keep up with the satellite acquisition. This is typically in the tens of millions of samples per second [26], [27] (with a spectral vector composed of one sample for each band), so we can consider as more than satisfactory a neural network labeling one to ten million spatial locations per second.

B. Self-Supervised Backbone

The main component of the neural network architecture is the backbone, which acts as a universal feature extractor. This feature extractor comprises a deep neural network that takes as input a multispectral image and computes a semantically meaningful representation composed of a number of features. This representation is then leveraged by further task-specific neural network heads for various applications.

The backbone feature extractor is designed and trained to produce features that can effectively be shared by all the heads; this means that the representations produced by the backbone must be general enough to adapt to a variety of possible vision tasks (such as classification, semantic segmentation, object detection, and more). This approach ensures the following two fundamental aspects within our architecture.

1) *Modularity:* The backbone is task-agnostic, thus it operates independently of the specific application heads or tasks we incorporate. This independence allows for separate training and functioning, promoting a modular framework where components can be adjusted or added without extensive restructuring.
2) *Efficiency:* The computational complexity is primarily concentrated within the backbone, performing the most resource-demanding computations just once. Subsequently, each head can execute its task in parallel, using these preprocessed features. This parallel execution enhances overall efficiency by minimizing redundant computations and optimizing task-specific processing.

In principle, an ideal backbone would be constituted by a foundational model [28] trained on vast amounts of data to generate highly general representations. While such foundational models are starting to emerge in the remote sensing literature [29], having one that is also lightweight remains elusive. One path toward a model of this kind is the use of SSL techniques that can exploit large datasets of unlabeled imagery and produce task-agnostic representations, coupled with an efficient design to match current computational capabilities of embedded systems.

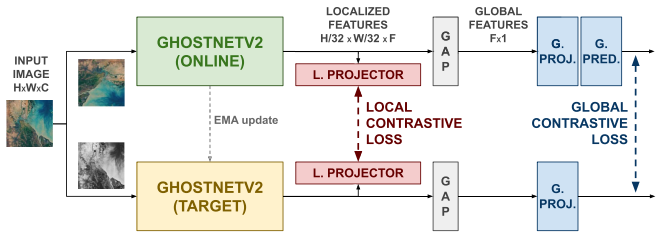


Fig. 2. SSL training of the GhostNetV2 backbone using local contrastive loss. In this framework, the online branch (upper) receives an input view of the image generated solely by applying spatial transformations, while the target branch (lower) processes a second view created by applying both spatial and intensity transformations. The structural configuration of the framework mirrors that presented in BYOL [5], augmented by the inclusion of a local contrastive branch. GAP is global average pooling.

Concerning SSL training, in this article, we chose to use the 590 326 Sentinel-2 images from the BigEarthNet dataset [30]. We propose to use an SSL technique that adapts the methodology outlined in BYOL [5], with a *local contrastive loss* inspired by the work of Islam et al. [7] to promote spatial features with a fine-grained resolution, useful for pixel-level tasks such as segmentation. A high-level overview of SSL training is depicted in Fig. 2. Two augmentations of an input multispectral image go through the online (top) and the target (bottom) networks, the latter being composed of weights obtained from a moving average of the weights of the online network. The projector layer is a linear operation on a spatially pooled representation of the entire image in a feature space. The online network has an extra linear layer called predictor. A global contrastive loss minimizes a dissimilarity metric between the output of the online predictor and the target projector. This global loss ensures that representation is globally semantically informative, and promotes clustering according to semantic classes for whole-image classification problems. However, it is not sufficient to ensure that the backbone learns fine-grained spatial features for segmentation problems. This is why a local contrastive loss is used to minimize pixelwise feature dissimilarity before spatial pooling. Overall, the SSL training loss is thus

$$L = \lambda L_{\mathrm{LC}} + (1-\lambda) L_{\mathrm{GC}} \tag{1}$$

with tradeoff parameter λ and

$$L_{\mathrm{LC}} = \frac{1}{|P|} \sum_{(p_c, p_{\mathrm{sc}}) \in P} \mathrm{NLL}(p_c, p_{\mathrm{sc}}) \tag{2}$$

$$L_{\mathrm{GC}} = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{||q_\theta(z_\theta)||_2 \cdot ||z'_\xi||_2} \tag{3}$$

with $q_\theta(z_\theta)$ the predictor's output of the online (upper) network and z_θ the output of the projection of the target network in Fig. 2. In (2), p_c are known points selected by defining a uniform $h \times w$ 2-D grid in the image that was augmented only by applying color transformations. Having defined the grid, and thus, the points, and knowing the spatial transformations applied to the other image, we can obtain the corresponding p_{sc} points in the second image, thus creating a $p_c - p_{\mathrm{sc}}$ point mapping between the pair of images. We use the latter to compute the negative log-likelihood as follows:

$$\mathrm{NLL}(p_c, p_t) = -\log \frac{e^{C'(p_c, p_{\mathrm{sc}})/\tau}}{\sum_{k \in \Omega_{\mathrm{sc}}} e^{C'(p_c, p_k)/\tau}} \tag{4}$$

where Ω_{sc} is the set of points p_{sc}, τ is a temperature hyper-parameter, and $C' \in R^{(h \times w) \times (h \times w)}$ is a dense correspondence map between the mapped points in the two images

$$C'(p_c, p_{\mathrm{sc}}) = \frac{F_\xi(p_c)^T F_\theta(p_{\mathrm{sc}})}{||F_\xi(p_c)||_2 ||F_\theta(p_{\mathrm{sc}})||_2} \tag{5}$$

where $F_\xi(p_c)$ and $F_\theta(p_{\mathrm{sc}})$ are the dense feature representation of the points obtained, respectively, from the target and online networks.

Concerning the architecture design, we suppose that four spectral bands (Red, Green, Blue, and Near Infrared) are used as input. The choice of bands is a tradeoff between the tasks to be solved and computational complexity: using all the available spectral bands could provide a more flexible backbone, but also increase computational complexity. For the sample tasks explored in this article, the red-green-blue (RGB) and near-infrared (NIR) bands provide adequate information, but other tasks could require additional bands. For instance, short-wave infrared (SWIR) could be useful to target fire detection and one could imagine to implement a slightly more complex head that takes as input the backbone features as well as the pixels of a new (e.g., SWIR) channel, and combines them to produce output for this specific task.

Given our primary goal of designing an efficient network, our approach to creating the backbone feature extractor is based on GhostNetV2 [31]. GhostNetV2 is a state-of-the-art lightweight CNN, specifically designed for fast inference on mobile and edge devices. As reported in [31], its performance surpasses that of MobileNetV3-L [32] by approximately 1%, achieving a top-1 accuracy of 77.8% in ImageNet classification. Notably, this achievement comes with a slightly increased number of floating point operation (FLOPs) compared to MobileNetV3's 355 millions of FLOPs (MFLOPs). Moreover, GhostNetV2 exhibits superior performance by approximately 2% over MobileViT-XS [33], despite MobileViT-XS has almost twice as many FLOPs as GhostNet. The main innovation of GhostNetV2 lies in realizing that conventional CNNs have highly redundant feature maps. Therefore, they can be obtained in a less expensive manner by initially generating a set of intrinsic feature maps, and then, using multiple cheap linear operations on them to derive the remaining redundant feature maps. This goal is achieved by a structure called GhostNet bottleneck, comprising stacked GhostNet modules, each incorporating a "hardware-friendly" attention mechanism known as decoupled fully connected attention. This attention mechanism aims to create feature maps that incorporate both local and long-distance information. Due to its extreme efficiency, combined with excellent performance, we selected GhostNetV2 as the backbone for feature extraction, excluding the four final layers specifically designed for classification. The output features to be used for the task-specific heads are taken at multiple depths of the GhostNetV2 architecture in order to provide a multiresolution feature bank, which is known [34],

[35] to be more effective than a single resolution for certain tasks. Specifically, feature maps after the fifth, seventh, and tenth layers at $\frac{1}{8}$, $\frac{1}{16}$, and $\frac{1}{32}$ of the input spatial resolution are selected.

Finally, we want to emphasize that the choice of GhostNetV2 is motivated by being the state-of-the-art model among low-complexity backbones at the time of writing, presenting an excellent tradeoff between complexity and accuracy, which allows us to verify if the onboard multitask AI system can achieve good performance. However, the considerations in this article are more general and GhostNetV2 could be replaced with any backbone providing multiresolution features, resulting in different tradeoffs between accuracy, latency, and power consumption.

C. Task-Specific Heads

The general framework outlined in this article enables a large variety of applications to be addressed thanks to the features extracted from the backbone. In order to evaluate the effectiveness of our design, we tested three tasks that can be relevant for onboard inference: cloud cover segmentation [36], floods segmentation [37], and marine littering whole-image multilabel classification [38]. The following datasets have been used for the sample tasks.

1) *Sentinel-2 Cloud Cover Segmentation Dataset* [36]: The dataset, developed by the Radiant Earth Foundation,[1] comprises 22 728 Sentinel-2 satellite images and their corresponding binary cloud masks. Each image has 512×512 pixels and represents imagery of a distinct area captured at a specific instance.
2) *Sen1Floods11* [37]: This dataset encompasses images from both Sentinel-1 and Sentinel-2 satellites, featuring binary masks distinguishing permanent water bodies from water associated with flood events. Focusing specifically on multispectral imagery, we filtered the dataset to retain solely the multispectral level 1C (L1C) images from Sentinel-2. This subset comprises only 446 images, each having 512×512 pixels.
3) *Marine Debris Archive (MARIDA)* [38]: MARIDA is a dataset for the classification of marine debris. The dataset includes 1381 Sentinel-2 multispectral images of 256×256 pixels, which distinguishes marine debris from various coexisting marine classes, including Sargassum macroalgae, ships, natural organic material, waves, wakes, foam, different water types (e.g., clear water, turbid water, sediment-laden water, and shallow water), and clouds. We use this dataset for whole-image multilabel classification where the entire input image is classified in one of 11 classes (different water types are aggregated into one class, as in the original paper, reducing the number from the original 15 to 11).

Three heads are, therefore, used in parallel in this example of multitask onboard inference. Since different kinds of tasks are to be solved, we designed two distinct low-complexity head types: one for multilabel classification and the other for segmentation.

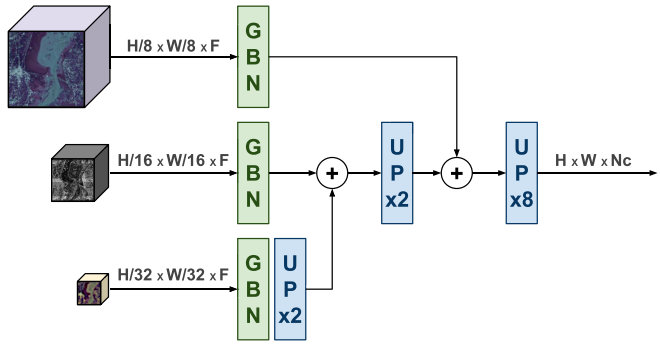


Fig. 3. Architecture of a segmentation head exploiting multiresolution features extracted by the backbone. UP is bilinear upsampling, and GBN is the GhostNetV2 bottleneck module [31].

1) Classification Head: The classification head consists of the four layers removed from the GhostNetV2 backbone, as detailed in Section III-B. It serves as a straightforward neural network comprising an initial global average pooling layer of only the features at the coarser spatial resolution, followed by a fully connected layer with rectified linear unit activation and a fully connected layer with Softmax activation.

2) Segmentation Head: Our segmentation head shares similarities with FCN-8s [39] or HRNet [34], [40], [41], as we adopted a similar multiresolution approach of aggregating feature sets extracted from various layers of the backbone. An overview is shown in Fig. 3. Following the efficiency paradigm, we utilized the GhostNet bottleneck module instead of traditional 2-D convolutions to reduce the number of parameters and FLOPs. This module is applied in parallel to the three resolutions, which are then added as residuals after bilinear upsampling.

The modularity of our architecture is evident: The backbone generates flexible, generalized features suited for multiple heads, each requiring finetuning solely on task-specific datasets. Introducing a novel task distinct from prior ones seamlessly integrates through the creation of a corresponding new head and its subsequent finetuning. Notably, finetuning the entire architecture (i.e., backbone and a single application head) is not desirable since the selective finetuning of the application heads, while keeping the backbone frozen, preserves the architecture's modularity and allows multiple tasks to be performed in parallel by the different heads. Furthermore, this approach requires a minimal amount of data for training, ensuring extremely fast finetuning of application heads.

D. Quantization

In addition to designing lightweight modules, neural network quantization [17] is also critical to improve memory requirements and inference speed. Our architecture underwent a post-training quantization process employing an 8-bit integer precision (INT8) scheme. In particular, static quantization is used to quantize both weights and activations and perform fully integer

[1] https://radiant.earth/

inference. INT8 is particularly optimized for implementation on embedded devices, and generally, but not always, suffers from small penalties in inference accuracy. A different, task-specific calibration dataset has been used for quantization of each head.

IV. Experimental Results

In this section, we initially validate the proposed design against state-of-the-art methods comparing their complexity as well as inference performance (in both single precision floating point (FP32) and INT8 precision) on the three sample tasks presented in Section III-C. Then, we use low-power hardware to analyze how the propose design fares in terms of total energy consumption and throughput. Finally, we present ablations by comparing a set of experiments featuring backbones of varying sizes and comparing how different SSL techniques affect the performance.

A. Implementation Details

The SSL training of the backbone spanned 550 epochs, employing a learning rate of 3×10^{-4} and a batch size of 3400 parallelized over 4 Nvidia A100 GPUs. After the completion of SSL training, we extracted the GhostNetV2 from the online branch and employed it as backbone feature extractor. Following a procedure similar to that presented in [7], we generated two distinct augmented views of a single image: I_s resulted solely from spatial transformations encompassing horizontal and vertical flips, random rotations $(90°, -90°, 180°, -180°)$, and random cropping. While, I_c was derived by applying a combination of spatial and spectral transformations, including color jittering, intensity manipulation, Gaussian blurring, and solarization. We determined that hyperparameter $\lambda = 0.1$ offered the optimal balance between the global and local contrastive losses. This choice resulted in a balanced performance across subpixel level and image-level tasks. Unless otherwise stated, the GhostNetV2 architecture uses multiplier $\alpha = 1.6$ in the choice of number of features. Finally, we specify that in order to perform network quantization in a fair way, the Intel neural compressor [42] library was used for all models. Each model was quantized in the same way, using a *static* configuration for posttraining quantization, with arithmetic entirely on 8-bit integers and calibrating each model with a calibration sample of 500 elements.

B. Model Comparison and Performance Evaluation

The tasks outlined in Section III-C include two semantic segmentation tasks (cloud cover segmentation [36] and flood segmentation [37]) and one classification task (marine litter [38] multilabel classification). Accordingly, we select some baselines and state-of-the-art models for segmentation and classification as terms of comparison. In particular, concerning segmentation we select DeepLabV3 [43] with MobileNetV3-L [32] backbone for a well-known and efficient baseline, HRNet18 [34] as the high-complexity state-of-the-art model, and UNet [44] with MobileVit-S as backbone [32] as a recent approach leveraging the representational power of Transformers, albeit with an eye to complexity. Concerning classification, we consider ResNet50 [45] as a standard baseline, MobileNetV3 [32] as a lightweight method, and ViT B-16 [46] as a high-complexity state-of-the-art model.

TABLE I
Model Complexity ($512 \times 512 \times 4$ Input)

Architecture	Params	MACs	FLOPs
Single-task Segmentation			
GhostNetv2 + Segmentation Head	9.55M	2.18G	4.47G
DeepLabV3 [44] + MobileNetV3-L [33]	11.02M	9.83G	19.74G
HRNet18 [35], [41], [42]	9.64M	18.39G	37.01G
UNet [45] + MobileViT-S [34]	8.04M	18.7G	37.63G
Single-task Classification			
GhostNetv2 + Classification Head	9.89M	2.09G	4.29G
MobileNetV3-L [33]	2.97M	1.12G	2.31G
ResNet50 [46]	23.52M	21.56G	43.29G
ViT B-16 [47]	86.61M	107.23G	214.75G
Multitask			
Proposed multitask	10.69M	2.28G	4.66G

A summary of the computational complexity of various methods is presented in Table I. For ease of comparison, we also include the proposed design with a single head alongside the multitask case. The proposed design demonstrates a very low number of FLOPs while maintaining a comparable number of parameters relative to other models, except for the large classification models, which generally require a larger number of parameters. In the context of classification, one might argue that the proposed method has more FLOPs than MobileNetV3, however, when considering the multitask scenario, including also inference for segmentation tasks, the efficient DeepLabV3 + MobileNetV3 framework has more FLOPs than our architecture. Thus, in the multitask setting, our proposed method remains the most efficient. Regarding the complexity of the individual application heads (excluding the backbone): our single segmentation head has 401K parameters and requires 92K multiply-accumulate (MACs), with 187M FLOPs for inference on a $512 \times 512 \times 4$ image provided to the backbone. Meanwhile, a single classification head has 743K parameters and requires 742K MACs, with 1.88M FLOPs for inference on a $512 \times 512 \times 4$ image provided to the backbone.

Heads were trained on a supervised way on each task dataset, without finetuning the backbone. In order to provide a fair comparison, the other methods were pretrained on ImageNet [47] and finetuned on the task datasets. We remark that freezing the backbone as dictated by our design goals is nevertheless penalizing compared to full finetuning. For this reason, we also report a benchmark in which the proposed model is fully finetuned for a specific task (this will be marked in the following as "SL"—supervised learning, in contrast to the "SSL" configuration for the frozen backbone), after the SSL pretraining.

Note that all tests conducted in the following section were performed using the original splits provided by the datasets to ensure that the results are comparable with those reported in the datasets' papers and associated benchmarks. The metrics

TABLE II
CLOUD COVER SEGMENTATION PERFORMANCE COMPARISON

Architecture	mIoU (FP32)	mIoU (INT8)	mF1 (FP32)	mF1 (INT8)
GhostNetv2 + Segmentation Head (SSL)	82.33	81.7	88.16	87.67
GhostNetv2 + Segmentation Head (SL)	83.75	83.41	89.13	88.85
DeepLabV3 + MobileNetV3-L	83.47	81.8	88.95	87.72
HRNet18	84.67	84.57	89.86	89.79
UNet + MobileViT-S	83.55	50.74	89.04	58.02

TABLE III
FLOODS SEGMENTATION PERFORMANCE COMPARISON

Architecture	mIoU (FP32)	mIoU (INT8)	mF1 (FP32)	mF1 (INT8)
GhostNetv2 + Segmentation Head (SSL)	40.32	39.78	50.45	49.66
GhostNetv2 + Segmentation Head (SL)	42.95	42.76	53.71	53.26
DeepLabV3 + MobileNetV3-L	41.03	34.32	51.39	43.62
HRNet18	54.68	54.62	65.72	65.51
UNet + MobileViT-S	59.65	14.75	70.25	19.18

TABLE IV
MARINE LITTERING CLASSIFICATION PERFORMANCE COMPARISON

Architecture	mF1 (FP32)	mF1 (INT8)
GhostNetv2 + Classification Head (SSL)	68.03	63.98
GhostNetv2 + Classification Head (SL)	61.24	60.38
MobileNetV3 Large	71.94	50.97
ResNet50	70.75	33.88
ViT B-16	64.22	63.74

presented in the following tables were calculated on the test set when available; otherwise, they were calculated on the validation set if the test set was not available in the dataset. The dataset configurations are as follows.

1) *Sentinel-2 Cloud Cover Segmentation Dataset:* Consists of 22 728 total images, with 11 748 in the training set and the remaining 10 980 in the test set.
2) *Sen1Floods11:* Consists of 426 total images, with 256 in the training set, 86 in the validation set, and 89 in the test set.
3) *MARIDA:* Consists of 1 381 images, with 694 in the training set, 328 in the validation set, and 359 in the test set.

Tables II–IV present performance comparisons for cloud cover segmentation, flood segmentation, and marine litter classification, respectively, while in Fig. 4, qualitative results are shown for cloud and flood segmentation tasks, comparing the different segmentation maps obtained from the models with ground truth and the corresponding RGB image. We evaluate the two segmentation tasks using the binary intersection-over-union (IoU) and the F1-Score, while for the classification task, we compute only the F1-score. We chose the binary IoU (hereafter referred to as "mIoU" for brevity) to ensure consistency with published results for cloud segmentation [48] and flood segmentation [37], aligning with the official metric used in both datasets. In addition, we compute the F1-Score for segmentation to provide a more detailed assessment of the model's performance across individual classes. In a sensitive task such as flood segmentation, it is particularly important to properly weigh the presence of false-positive pixels incorrectly labeled as "Flood." A significant number of false positives could suggest a nonexistent flood zone, potentially leading to unwanted triggers. Thus, relying solely on mIoU does not provide sufficient information about false positives, making the F1-Score an important complementary metric.

As shown in Table II, it is noteworthy that all models exhibit relatively similar performance in clouds segmentation, within about 2% points of variation in mIOU between the best and worst, both in FP32 and INT8 quantization, with the exception of UNet + MobileViT-S, which suffers greatly from quantization. Furthermore, it is interesting to notice that the SL version of our architecture is only marginally better than the SSL version, suggesting that the design constraint of freezing the backbone may not have a big impact. The differences in performance between the models is even smaller if we look at the F1-Score, indicating again how well-known state of the art (SOTA) models with higher performance do not have excessive gains in quality metrics compared to more efficient models such as MobileNetV3 and our proposed architecture.

The results on floods segmentation, shown in Table III, show that the higher complexity models are generally superior in this specific task, while the proposed architecture provides better performance than the direct low-complexity alternative (DeepLabV3 + MobileNetV3-L). Indeed, scaling experiments reported in Table VIII suggest that a larger model would improve performance in exchange of speed.

Finally, the marine littering task addressed a problem of multilabel classification instead of segmentation, providing an analysis of how well the proposed model can address heterogeneous tasks that both need fine and coarse grained features. In this task, we notice that the proposed method is very close to the best FP32 results, and it is the best overall in INT8. It is worth remarking that the MARIDA dataset is very small, leading some highly complex methods to overfit when finetuned.

C. Analysis on Low-Power Hardware

In order to validate the suitability of the proposed design for onboard usage, we performed some tests on an Nvidia Jetson Orin Nano 8GB embedded system. While not currently space-qualified, it is a low-power hardware platform with a CPU and

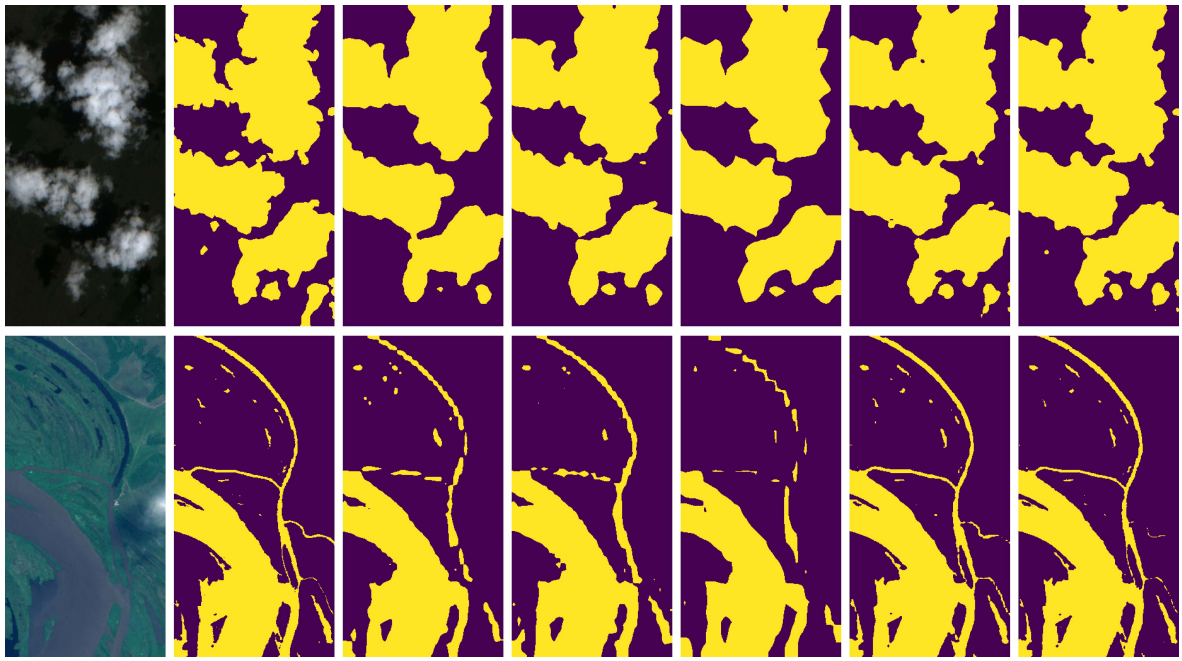


Fig. 4. Qualitative comparison of different segmentation models. Top row: Cloud cover segmentation; Bottom row: Flood segmentation. Left to right: RGB image, Ground Truth, GhostNetv2 + Segmentation Head (SSL), GhostNetv2 + Segmentation Head (SL), DeepLabV3 + MobileNetV3-L, HRNet18, and UNet + MobileViT-S.

TABLE V
THROUGHPUT AND POWER CONSUMPTION ON LOW-POWER HARDWARE

Model	Tasks	Lat. (FP32)	Lat. (INT8)	Pwr-Norm. Lat. (FP32)	Pwr-Norm. Lat. (INT8)	Avg Pwr
GhostNetv2 + 3 parallel heads (**Ours**)	SSC	56.77 ms	34.67 ms	48.66 ms	29.72 ms	6.0 W
DeepLabV3 + MobileNetV3-L (**DLMN**)	S	39.52 ms	15.70 ms	32.75 ms	13.01 ms	5.8 W
HRNet18 (**HR**)	S	118.07 ms	47.72 ms	106.26 ms	42.95 ms	6.3 W
UNet + MobileViT Small (**mViT**)	S	117.79 ms	82.94 ms	109.38 ms	77.02 ms	6.5 W
MobileNetV3 Large (**MN**)	C	18.61 ms	9.19 ms	15.59 ms	7.61 ms	5.8 W
ResNet50 (**RN50**)	C	45.98 ms	15.00 ms	42.04 ms	13.71 ms	6.4 W
ViT B-16 (**ViT**)	C	364.79 ms	296.27 ms	343.94 ms	279.34 ms	6.6 W

GPU for AI acceleration with a peak power budget of 7 W or 15 W, depending on usage mode, which allows us to characterize latency, total energy consumption, as well as limitations in image size due to memory on a sufficiently representative system. All tests are conducted in the 7-W board mode.

Table V reports some results for a $512 \times 512 \times 4$ input in terms of inference latency, average power consumption and power-normalized latency. The latter is computed as the product between latency and average power normalized by 7 W, i.e., the maximum power budget of the system. It should be noticed that average power consumption serves as a validation of whether the method is fully utilizing the available resources, by staying close to the 7-W budget, or not. Latency results are averaged over ten runs with 200 warmup iterations. We compare the proposed multitask architecture with the aforementioned baseline and state-of-the-art architectures for individual tasks. We can notice that the proposed design can solve three tasks [two segmentation (S) and one classification (C)] with a latency that is inferior of several other single-task models. We also notice that INT8 quantization provides almost a factor of 2 speedup. Considering the input resolution has 512×512 spatial locations to be labeled, we can say that the FP32 inference time of 56.77 ms corresponds to a throughput of 4.62 megapixel/s and that the INT8 inference time of 34.67 ms corresponds to a throughput of 7.56 megapixel/s.

Table VI presents a tradeoff analysis for the multitask problem. In this analysis, we investigate what is the total energy consumed to solve the three tasks, as a function of latency and instantaneous power, in relation to the inference quality. The proposed design is compared with different combinations of methods to address the three tasks. These methods need to be run serially as the system is already fully used by each single task. In particular, we select some interesting combinations including fastest baseline (DLMN + DLMN + MN, refer to Table V for acronyms), highest FP32 quality (HR + mViT + MN), highest INT8 quality (HR + HR + ViT), Transformers-only (mViT + mViT + ViT), and CNN-only (DLMN + DLMN + RN50). Taking the proposed design as the reference, we report E_Δ as the percentage difference between the energy in Joules consumed to complete the three tasks, and Q_Δ as the average percentage difference in inference metrics (i.e., the percentage difference in mIOU or F1 is computed for each task, and then,

TABLE VI
ENERGY-QUALITY TRADEOFF ON LOW-POWER HARDWARE

Method	Latency (FP32)	E_Δ (FP32)	Q_Δ (FP32)	Latency (INT8)	E_Δ (INT8)	Q_Δ (INT8)
Ours	56.77 ms	0%	0%	34.67 ms	0%	0%
DLMN + DLMN + MN	97.65 ms	+66.26%	+2.82%	40.58 ms	+13.16%	−7.94%
HR + mViT + MN	254.46 ms	+374.80%	+8.68%	139.85 ms	+329.32%	−14.58%
HR + HR + ViT	600.92 ms	+1043.50%	+3.46%	357.72 ms	+1129.06%	+5.59%
mViT + mViT + ViT	600.36 ms	+1056.29%	+4.45%	462.15 ms	+1358.31%	−21.73%
DLMN + DLMN + RN50	125.01 ms	+120.95%	+2.23%	46.39 ms	+33.68%	−16.84%

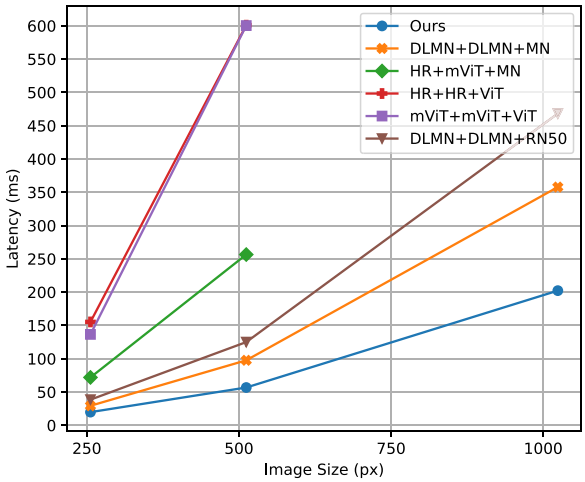


Fig. 5. Inference times using different image sizes on FP32 models. Pipelines that include mViT or ViT have no latency for 1024×1024 images because there is insufficient RAM on the Jetson system to run them.

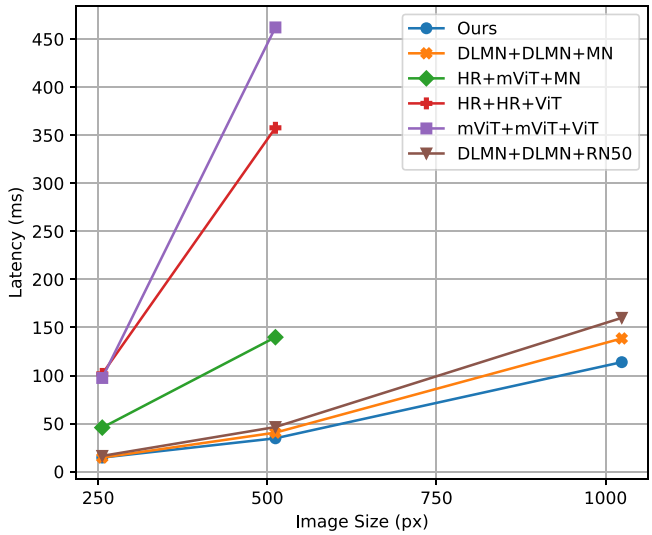


Fig. 6. Inference times using different image sizes on INT8 models. Pipelines that include mViT or ViT have no latency for 1024× 1024 images because there is insufficient RAM on the Jetson system to run them.

averaged over the three tasks). We are not surprised that because of the parallel multitask approach, the proposed design requires the least energy to complete the tasks. While it does not provide the best quality overall, modest improvements in quality are offset by large increases in energy consumption (e.g., +8% quality requires +374% energy with HR + mViT + MN), highlighting the good tradeoff achieved by the proposed method.

Finally, we present how the methods scale as a function of image size in Figs. 5 and 6. The upper limit in image size, dictated by the Jetson's 8-GB shared system memory, is 1024 × 1024 for all methods except Transformers, which run out of memory at this resolution. Generally, slightly better efficiency is achieved with a 1024 × 1024 input, reaching 8.91 Megapixels/s of INT8 throughput compared to 7.56 Megapixels/s for a 512 × 512 input. Indeed, real onboard acquisitions may be significantly larger than 512 × 512 or 1024 × 1024. However, a tiling strategy would be adopted onboard, where the large image would be partitioned into tiles as large as the system memory allows. This is also why we present results in terms of throughput, which normalizes latency by image size.

D. Backbone Architecture Ablation

The choice of using GhostNetV2 as the backbone architecture for our main experiments was driven by being the state-of-the-art model among low-complexity architectures. However, one might wonder how a different backbone compares to GhostNetV2 under the specific SSL multitask setting under study. For this purpose, we test MobileNetV3-L as an alternative and present the results in Table VII. It can be noticed that MobileNet offers a different tradeoff between accuracy and latency, being worse on the former and faster for the latter. Whether this is desirable, it depends on the specifics of the mission under design, and in particular, its speed target.

For a fair comparison, both GhostNetV2 and MobileNetV3-L were trained under identical conditions: the SSL pretraining was conducted for the same number of epochs on the same dataset, using a width multiplier of 1.6 for both networks. Furthermore, the application-specific heads employed for evaluation were consistent with those described in Section III-C.

E. Backbone Size Ablations

As mentioned previously, all experiments conducted on a GhostNetV2 backbone with the α width parameter set to 1.6. Since this parameter influences the number of features in different layers of the network and the input channels for the various heads, we conducted experiments to explore how the network's parameters, MACs, FLOPs, and performance change with varying α values, reported in Table VIII. In this experiment, SSL training for the backbone feature extractor for 100 epochs is followed by supervised training of the segmentation head for the cloud cover task. It is interesting to notice that the model scales beyond the $\alpha = 1.6$ value used in all our experiments with

TABLE VII
BACKBONE ABLATION

Backbone	Cloud (mIOU)	Floods (mIOU)	Marine Litter (mF1)	Latency
GhostNetV2 (SSL) + 3 parallel heads	82.33	40.32	68.03	56.77 ms
MobileNetV3-L (SSL) + 3 parallel heads	81.45	40.00	64.78	32.51 ms

TABLE VIII
MODEL SIZE ABLATION FOR CLOUD COVER SEGMENTATION

width α	Params	MACs	FLOPs	mIoU (FP32)	mIoU (INT8)
1	3.78M	912.28K	1.89G	79.86	79.54
1.6	9.55M	2.18G	4.47G	80.93	80.19
2	14.69M	3.34G	6.81G	81.83	81.05

TABLE IX
SSL TRAINING ABLATION FOR CLOUD COVER SEGMENTATION

SSL Training	mIoU (FP32)
Local Contrastive Loss (ours)	82.33
SSL Remote Sensing [50]	79.16

improved mIOU. However, complexity also scales accordingly, so, while we found $\alpha = 1.6$ to be a good tradeoff, if a mission desires it can sacrifice some speed for higher quality maintaining the proposed design by choosing $\alpha = 2$ or higher.

F. Comparison of SSL Methods

We conducted a comparative analysis of our local contrastive loss technique, detailed in Section III-B, against the method presented in [12], which was specifically tailored for contrastive learning training on remote sensing RGB images and can be considered a state-of-the-art self-supervised training approach in the remote sensing field.

In order to provide a fair assessment of the training procedure, we used the same GhostNetV2 backbone architecture of our main experiments. The original authors code was adapted to handle transformations over four spectral channels, while we preserved the original hyperparameters and experimental settings as in the original work, pretraining the backbone on BigEarthNet dataset. Then, we froze the weights of our GhostNetV2 backbone and conducted a comprehensive evaluation on the cloud cover dataset, replicating the methodology outlined in Section IV-B. The results are summarized in Table IX and clearly demonstrate the substantial performance enhancement achieved through our SSL pretraining compared via a combination of local and global contrastive loss, with respect to the methodology proposed in [12].

V. CONCLUSION

In this article, we presented a high-level conceptualization of how to design an AI payload for a spacecraft capable of addressing multiple tasks of interest directly onboard to provide rapid response to events or improved system functionality. We also delved into a low-complexity architecture and its training process, leveraging SSL to enable a modular approach as well as reduce requirements for labeled data. Extensive experiments over three tasks of interest on low-power hardware show that the proposed method is capable of inference quality close to that of high-complexity state-of-the-art models at a fraction of energy consumption. Moreover, we measured a high absolute throughput that would make real-time operations feasible.

REFERENCES

[1] 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.

[2] M. Kerr et al., "Eo-alert: A satellite architecture for detection and monitoring of extreme events in real time," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2021, pp. 168–171.

[3] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[4] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9729–9738.

[5] J.-B. Grill et al., "Bootstrap your own latent: A new approach to self-supervised learning," in *Proc. Neural Inf. Process. Syst.*, 2020, pp. 21271–21284.

[6] K. He, R. Girshick, and P. Dollár, "Rethinking imagenet pre-training," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2018, pp. 4918–4927.

[7] A. Islam, B. Lundell, H. Sawhney, S. N. Sinha, P. Morales, and R. J. Radke, "Self-supervised learning with local contrastive loss for detection and semantic segmentation," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis.*, 2023, pp. 5624–5633.

[8] A. Ziegler and Y. M. Asano, "Self-supervised learning of object parts for semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 14502–14511.

[9] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li, "Dense contrastive learning for self-supervised visual pre-training," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3024–3033.

[10] C. Tao, J. Qi, W. Lu, H. Wang, and H. Li, "Remote sensing image scene classification with self-supervised paradigm under limited labeled samples," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, 2022, Art. no. 8004005.

[11] J. Kang, R. Fernandez-Beltran, P. Duan, S. Liu, and A. J. Plaza, "Deep unsupervised embedding for remotely sensed images based on spatially augmented momentum contrast," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 3, pp. 2598–2610, Mar. 2021.

[12] W. Li, H. Chen, and Z. Shi, "Semantic segmentation of remote sensing images with self-supervised multitask representation learning," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 14, pp. 6438–6450, 2021.

[13] A. Montanaro, D. Valsesia, G. Fracastoro, and E. Magli, "Semi-supervised learning for joint SAR and multispectral land cover classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 19, 2022, Art. no. 2506305.

[14] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[15] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6848–6856.

[16] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[17] B. Jacob et al., "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2704–2713.

[18] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks," *Adv. Neural Inf. Process. Syst.*, vol. 29, 2016.

[19] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015, *arXiv:1503.02531*.

[20] B. Zoph and Q. Le, "Neural architecture search with reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2016.

[21] 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.
[22] Y. Yao, Z. Jiang, H. Zhang, and Y. Zhou, "On-board ship detection in micro-nano satellite based on deep learning and COTS component," *Remote Sens.*, vol. 11, no. 7, 2019, Art. no. 762.
[23] M. Ziaja et al., "Benchmarking deep learning for on-board space applications," *Remote Sens.*, vol. 13, no. 19, 2021, Art. no. 3981.
[24] V. Růžička et al., "Ravæn: Unsupervised change detection of extreme events using ML on-board satellites," *Sci. Rep.*, vol. 12, no. 1, 2022, Art. no. 16939.
[25] M. Cilia, N. Prette, E. Magli, B. Sang, and S. Pieraccini, "Onboard data reduction for multispectral and hyperspectral images via cloud screening," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2020, pp. 6230–6233.
[26] L. Santos, L. Berrojo, J. Moreno, J. F. López, and R. Sarmiento, "Multispectral and hyperspectral lossless compressor for space applications (HyLoC): A low-complexity FPGA implementation of the CCSDS 123 standard," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 2, pp. 757–770, Feb. 2016.
[27] J. Fjeldtvedt, M. Orlandić, and T. A. Johansen, "An efficient real-time FPGA implementation of the CCSDS-123 compression standard for hyperspectral images," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 10, pp. 3841–3852, Oct. 2018.
[28] M. Awais et al., "Foundational models defining a new era in vision: A survey and outlook," 2023, *arXiv:2307.13721*.
[29] J. Jakubik et al., "Foundation models for generalist geospatial artificial intelligence," 2023, *arXiv:2310.18660*.
[30] G. Sumbul, M. Charfuelan, B. Demir, and V. Markl, "BigEarthNet: A large-scale benchmark archive for remote sensing image understanding," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2019, pp. 5901–5904.
[31] Y. Tang, K. Han, J. Guo, C. Xu, C. Xu, and Y. Wang, "GhostNetV2: Enhance cheap operation with long-range attention," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 9969–9982, 2022.
[32] A. Howard et al., "Searching for mobileNetV3," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 1314–1324.
[33] S. Mehta and M. Rastegari, "MobileViT: Light-weight, general-purpose, and mobile-friendly vision transformer," in *Proc. Int. Conf. Learn. Representations*, 2021.
[34] K. Sun et al., "High-resolution representations for labeling pixels and regions," 2019, *arXiv:1904.04514*.
[35] D. Valsesia and E. Magli, "Super-resolved multi-temporal segmentation with deep permutation-invariant networks," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2022, pp. 995–998.
[36] Radiant Earth Foundation, "Sentinel-2 cloud cover segmentation dataset (version 1)," 2022. [Online]. Available: https://doi.org/10.34911/rdnt.hfq6m7
[37] D. Bonafilia, B. Tellman, T. Anderson, and E. Issenberg, "Sen1Floods11: A georeferenced dataset to train and test deep learning flood algorithms for Sentinel-1," in *Proc. 2020 IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2020, pp. 835–845.
[38] K. Kikaki, I. Kakogeorgiou, P. Mikeli, D. E. Raitsos, and K. Karantzalos, "MARIDA: A benchmark for marine debris detection from Sentinel-2 remote sensing data," *PLoS One*, vol. 17, no. 1, pp. 1–20, 2022. [Online]. Available: https://doi.org/10.1371/journal.pone.0262247
[39] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. Comput. Vis. Pattern Recognit. Conf.*, 2016, pp. 3431–3440.
[40] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 5693–5703.
[41] J. Wang et al., "Deep high-resolution representation learning for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 10, pp. 3349–3364, Oct. 2021.
[42] F. Tian, H. Chang, H. Shen, and S. Chen, "Intel neural compressor," 2022. [Online]. Available: https://github.com/intel/neural-compressor
[43] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*.
[44] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. 18th Int. Conf. Med. Image Comput. Comput.-Assist. Interv.*, Munich, Germany, Oct., 5–9, 2015, pp. 234–241.
[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[46] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. Int. Conf. Learn. Representations*, 2020.
[47] O. Russakovsky et al., "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.
[48] Driven Data, "On cloud N: Cloud cover detection challenge," 2023. [Online]. Available: https://github.com/drivendataorg/cloud-cover-winners



Gabriele Inzerillo received the M.Sc. degree in computer engineering in 2022 from the Politecnico di Torino, Turin, Italy, where he is currently working toward the Ph.D. degree in deep learning for remote sensing, under a collaboration between Politecnico di Torino and the European Space Agency, within the Open Space Innovation Platform (OSIP) Framework.



Diego Valsesia (Member, IEEE) received the Ph.D. degree in electronic and communication engineering from the Politecnico di Torino, Turin, Italy, in 2016.

He is currently an Assistant Professor with the Department of Electronics and Telecommunications (DET), Politecnico di Torino. His main research interests include processing of remote sensing images and deep learning for inverse problems in imaging.

Dr. Valsesia is an Associate Editor for IEEE TRANSACTIONS ON IMAGE PROCESSING, for which he received the 2023 Outstanding Editorial Board Member Award. He is a Member of the EURASIP Technical Area Committee for Signal and Data Analytics for Machine Learning and a Member of the European Laboratory for Learning and Intelligent Systems (ELLIS) society. He was the recipient of the IEEE International Conference on Image Processing (ICIP) 2019 Best Paper Award, and the IEEE Multimedia 2019 Best Paper Award.



Enrico Magli (Fellow, IEEE) received the M.Sc. and Ph.D. degrees in electronic engineering from the Politecnico di Torino, Turin, Italy, in 1997 and 2001, respectively.

He is currently a Full Professor with the Politecnico di Torino, where he leads the Image Processing and Learning group, performing research in the fields of deep learning for image and video processing, image compression, and image forensic for multimedia and remote sensing applications.

Prof. Magli is an Associate Editor for IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY and the *EURASIP Journal on Image and Video Processing*. He is a Fellow of the ELLIS Society for the advancement of artificial intelligence in Europe, and has been an IEEE Distinguished Lecturer from 2015 to 2016. He was the recipient of the IEEE Geoscience and Remote Sensing Society 2011 Transactions Prize Paper Award, the IEEE ICIP 2015 Best Student Paper Award (as senior author), the IEEE ICIP 2019 Best Paper Award, the IEEE Multimedia 2019 Best Paper Award, and the 2010 and 2014 Best Associate Editor Award of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY.