remote sensing

MDPI

Article

Cloud Detection Autonomous System Based on Machine Learning and COTS Components On-Board Small Satellites

Carlos Salazar [1,†], Jesus Gonzalez-Llorente [1,*,†], Lorena Cardenas [1,2], Javier Mendez [1,2], Sonia Rincon [1,2], Julian Rodriguez-Ferreira [3] and Ignacio F. Acero [4]

[1] Centro de Investigación en Tecnologías Aeroespaciales (CITAE), Cra 8 # 58-67, Cali 760011, Colombia
[2] Fuerza Aérea Colombiana (FAC), Cra 8 # 58-67, Cali 760011, Colombia
[3] Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones, Universidad Industrial de Santander, Cl 9 Cra 27 Ciudad Universitaria, Bucaramanga 680002, Colombia
[4] Escuela de Ciencias Exactas e Ingeniería, Universidad Sergio Arboleda, Calle 74 # 14-14, Bogota 110221, Colombia
* Correspondence: jdgonzalezl@ieee.org
† These authors contributed equally to this work.

Abstract: One of the main applications of small satellites is Earth observation. CubeSats and different kinds of nanosatellites usually form constellations that obtain images mainly using an optical payload. There is a massive amount of data generated by these satellites and a limited capacity of download due to volume and mass constraints that make it difficult to use high-speed communication systems and high-power systems. For this reason, it is important to develop satellites with the autonomy to process data on board. In this way, the limited communication channel can be used efficiently to download relevant images containing the required information. In this paper, a system for the satellite on-board processing of RGB images is proposed, which automatically detects the cloud coverage level to prioritize the images and effectively uses the download time and the mission operation center. The system implements a Convolutional Neural Network (CNN) on a Commercial off-the-Shelf (COTS) microcontroller that receives the image and returns the cloud level (priority). After training, the system was tested on a dataset of 100 images with an accuracy of 0.9 and it was also evaluated with CubeSat images to evaluate the performance of a different image sensor. This implementation contributes to the development of autonomous satellites with processing on board.

Keywords: machine learning; TinyML; convolutional neural networks; on-board processing; edge processing; CubeSat; small satellites; AI on the edge

Citation: Salazar, C.; Gonzalez-Llorente, J.; Cardenas, L.; Mendez, J.; Rincon, S.; Rodriguez-Ferreira, J.; Acero, I.F. Cloud Detection Autonomous System Based on Machine Learning and COTS Components On-Board Small Satellites. *Remote Sens.* **2022**, *14*, 5597. https://doi.org/10.3390/rs14215597

Academic Editors: Angelo Cervone, Vaios Lappas and Vittorio Franzese

Received: 12 September 2022
Accepted: 2 November 2022
Published: 6 November 2022

1. Introduction

Modern small satellites are having a significant impact on our world today, as they are being introduced in education, commercial applications, and institutions from developing countries, due to the incorporation of Commercial off-the-Shelf (COTS) components and rapid development cycles [1–3]. The physical limitation of pico/nano/microsatellites has not been an impediment to the development of missions for technology demonstration, science, communication, and Earth observation [4–8]. Earth observation is one of the main applications of these modern small satellites [8–10]. The first demonstrations of small satellites for Earth observation occurred around the year 2000, and their commercial viability was achieved with the formation of constellations that increased the temporal resolution and were possible due to their low cost [1]. The low cost was achieved after hardware miniaturization promoted the development of Earth observation payloads for CubeSats with characteristics ranging from panchromatic to hyperspectral images, depending on the physical limitations of volume and weight [10,11]. Small satellites are used for wildfire detection, deforestation monitoring, ship detection, land use, surface water mapping, among other applications using optical sensors. The combination of different bands can highlight

features that can be easily recognized. However, cloud-covered images limit most of these applications because clouds tend to block the visible wavelength. Although cloudy images are useful for studying the Earth cloud system, these images can be a problem for many Earth observation applications.

The physical dimensions of small satellites limit the communication link and the available power, making it difficult to download or process all the generated data. Many downloaded images are useless because they are covered by clouds, whereas more images are captured and a high volume of data is generated in orbit. One way to address this problem is to download reduced size versions of images (thumbnails) to inspect the utility of the image, as carried out by the FACSAT-1 mission [12]. In this case, it is common to spend more than five minutes downloading a thumbnail and evaluating whether it is worthwhile to download the entire image according to the cloud coverage percentage. Considering that the communication window is around 10 min, the operator spends 50% of the operating time on this. Another way to address the problem is to automate the on-board processing of the image for cloud coverage calculation using artificial intelligence (AI). However, the application of AI in modern small satellites is limited by the processing speed and power consumption of the on-board computers.

Recently, the application of artificial intelligence (AI) to the edge in space systems has attracted the interest of researchers [13]. The application of machine learning (ML) in resource-constrained environments has promoted the development of hardware platforms and frameworks to enable Deep Learning techniques such as Convolutional Neural Networks (CNN) in small computers [14]. For example, CNN applications to cloud segmentation in multispectral data was proposed using GPU [15]. Most of the hardware solutions for Deep Learning in edge processing are based on Application-Specific Integrated Circuit (ASIC), Graphics Processing Unit (GPU) or Field Programmable Gate Arrays (FPGA) [16], which are high-processing tools that require high power. Therefore, these hardware solutions are not common on CubeSats or other small satellites due to their limited power capabilities.

To enable AI on-board CubeSat, recent efforts have focused on implementing Deep Learning for traditional microcontrollers. A scheme based on Deep Learning for on-board ship detection was proposed in [17]. The application of on-board processing using CNN for selecting an image to download has been proposed in [18]. Another application of CNN for wildfire image classification on board CubeSat is presented in [19]. However, the accuracy of these implementations is usually lower than that of the implementation in specific hardware solutions due to quantization and pruning techniques that enable edge in space systems.

In order to implement satellite on-board processing, a CNN for cloud detection in hyperspectral images, CloudScout, was tested on a Vision Processing Unit (VPU), a COTS hardware accelerator for Deep Learning [20]. CloudScout uses three bands to categorize the images as cloudy or non-cloudy. Interest in applying CNN on board small satellites has encouraged optimization techniques and model simplification to demonstrate its suitability for COTS low-power hardware. The combination of image compression and a lightweight version of U-net [21], LU-net, was proposed to detect clouds using four bands (R,G,B and nearIR); this was tested on the Raspberry Pi platform (ARMv8 CPU) [22]. A variant of U-net, RS-Net, was proposed and evaluated with different combinations of bands available in Landsat 8 images [23]. It achieved a significant performance using only the RGB image, which is of great relevance for nanosatellites without a multispectral sensor with low power and computation capabilities. With this approach, a reduced version of RS-net, NU-Net, was proposed for cloud segmentation using RGB images and was tested on an ARM-Cortex M4 microcontroller, which was used for image priorization [24].

This paper details an automatic image priorization system based on cloud coverage which can be installed on-board a small spacecraft such as Cubesat—pico/nano/microsatellites. The implementation of this system can detect the cloud coverage in an RGB image; in this way, during the operation it is not necessary to download the thumbnail image to

evaluate the cloud coverage. The system can be used to provide the cloud coverage index of a set of images, which can be ordered accordingly. Cloud detection was obtained by a Convolutional Neural Network running on a microcontroller and was trained by using a dataset from Sentinel-2 which was divided into training and test datasets. The accuracy obtained with the test dataset of 100 images was 0.91 and the F1-score was 0.83. It is known that CNN has some problems with unseen domains. A change in the imager or the orbit of the satellite changes the domain of the images. For this reason, we also tested the system with a dataset from an Earth observation 3U CubeSat.

The main contributions reported in this paper are the following:

1. A proposal for an autonomous system to be used on board small satellites to prioritize images according to cloud coverage;
2. Demonstration of CNN-based cloud detection on a COTS-based embedded system not optimized for Deep Learning and using an open source framework;
3. Evaluation of the effect of quantization of CNN performance by comparing the results of the embedded system against the PC;
4. Demonstration of CNN performance in the unseen domain by testing the systems with a dataset from a different imager—the images obtained from a CubeSat mission.

2. Methods and Materials

To detect the cloud coverage of images on board CubeSats, we proposed a module capable of detecting the clouds and quantifying the percentage of cloud coverage. The module can be connected to a master device, and it receives the image and returns the cloud coverage calculated from a cloud mask generated by a CNN. The cloud coverage is used to order the image accordingly during image prioritization. In order to develop this module we required a CNN architecture that processes the image to detect the cloud, a cloud coverage calculation, a dataset to train and test the CNN, and the implementation of the module on an embedded device suitable for CubeSats. All these elements are described in this section along with the evaluation process to verify the module.

2.1. Cloud Detection Using the CNN

Cloud detection is the core of the module and is based on the specific CNN architecture, NU-Net, proposed by Park [24]. This architecture was chosen because its potential for cloud detection in RGB images was demonstrated in [24] by comparing it against several Deep Learning approaches such LU-net [22], RS-Net [23] and classical techniques such as Random Forest Classifier (RFC), Gradient Boosting Classifier (GBC) and Support Vector Machine (SVM), as well as its suitability to be implemented in a microcontroller. This architecture for segmentation tasks was selected over a purely regression architecture because it states in a clearer way, making it pixel-wise. NU-Net is a CNN based on RS-Net that is a simplified version of U-Net. NU-Net consists of a multistream architecture with a reduced number of kernels to make the networks suitable for microcontrollers [24]. The two streams are a single-band stream and a multi-band stream. The single-band stream is focused on extracting spatial features with a larger network than the multi-band stream; both networks are concatenated at the end of the network to obtain an image of the same size of the input image as shown in Figure 1. More details of NU-Net can be found in Park [24]. In this work, we have explored a variation of this architecture, NU-Net-mod. With this variation, we were looking for a reduction in the size of the network by decreasing the number of weights. To achieve this objective, we maintained the multistream approach; however, both streams were single band, and the input image was transformed into a single band image before splitting, as shown in Figure 1.

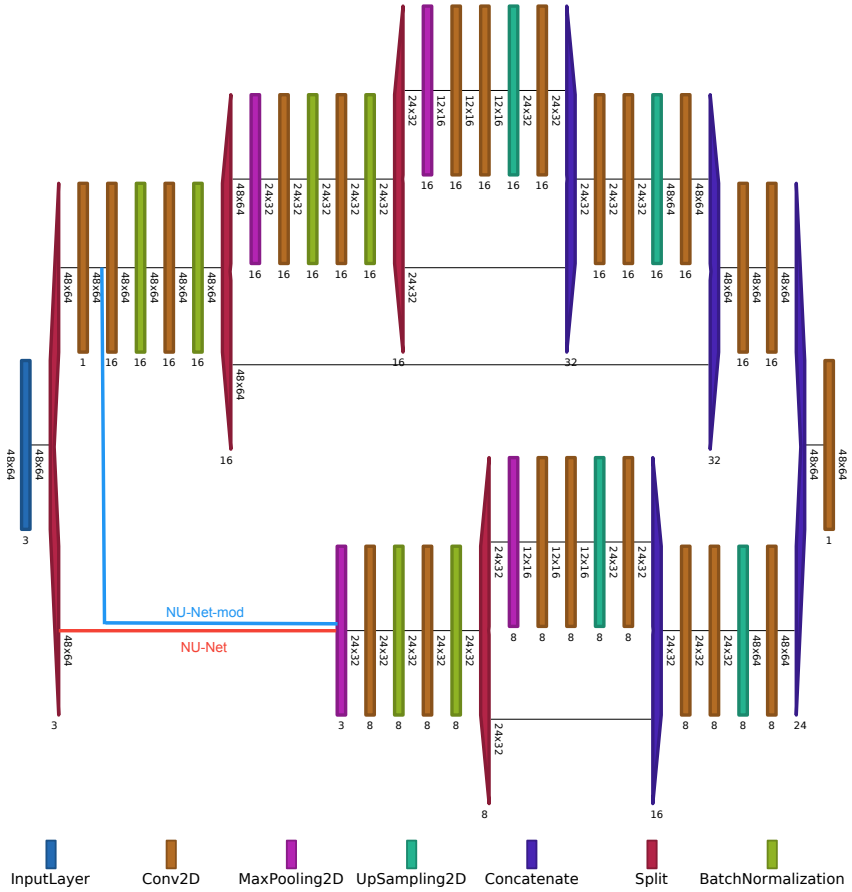


Figure 1. Architecture of NU-Net [24] and Nu-Net-mod visualized using [25]. The blue and red lines show the input of the second stream in NU-Net-mod and NU-Net respectively. The depth of the feature map is denoted at the bottom of each box.

2.1.1. Dataset Description

Two datasets from different satellites were used as follows:

The first dataset, used for the NU-Net training and test, consisted of 600 RGB images in true color and low resolution (60 m) from Sentinel-2B that were resized to 64 × 48 pixels and 8-bits. There was a cloud mask for each image of the same dimensions (64 × 48 pixels), and it was divided into a training dataset and a test dataset with 300 images each, as explained by Park [24]. This dataset resembles the kind of image acquired by nanosatellites and contains the labels required for training the CNN. This dataset was downloaded from the supplementary material provided by [24].

The second dataset, used only for evaluation under unseen domain, was made up of a set of thumbnail images acquired by the FACSAT-1 nanosatellite. The thumbnail images in this dataset are RGB images of 640 × 480 pixels which were resized to 64 × 48 pixels to match with the train dataset. As described in the Introduction, these are the images used by the operators of the nanosatellite to evaluate the cloud coverage of the image in order to download a higher-resolution image. There is no label for this dataset; however, these images were useful for evaluating the generalization of the CNN and its performance for an unseen domain because these images were acquired by a different imager. The method of qualitative evaluation using this dataset is explained at the end of this section.

2.1.2. Training Setup and Evaluation Metrics for the CNN

The architecture of NU-net was created with Python using Tensorflow and the Keras Deep Learning framework. CNN was trained on a PC using 100 randomly selected images from the training dataset and data augmentation was used to obtain a more generalized

model and reduce overfitting. The data augmentation was performed with the parameters listed in Table 1.

Table 1. Summary of parameters used for data augmentation.

Parameter	Value
Rotation range	0.2
Width shift range	0.05
Height shift range	0.05
Shear range	0.05
Zoom range	0.05
Horizontal flip	True
Fill mode	nearest neighbor

For the training process, the selected parameters are listed in Table 2.

Table 2. Summary of parameters used for training.

Parameter	Value
Loss function	binary crossentropy
Optimizer	Adam
Learning rate	1×10^{-4}
Number of batches	10
Steps per epoch	200
Epochs	100

To evaluate the performance of NU-Net, quantitative metrics were calculated using the first dataset, which contained the cloud mask used for label. Accuracy, precision, recall, and $F1$ were calculated based on the confusion matrix on all pixels in the dataset as proposed by Jepppensen [23]. Using the four metrics, misinterpretations were avoided due to imbalanced classes in the distribution of the dataset. The following describes how each metric was calculated:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = 2 * \frac{precision * recall}{precision + recall}, \tag{4}$$

where TP, TN, FP, and FN are true positive, true negative, false positive and false negative, respectively.

In addition to the metrics that evaluated the performance of the CNN pixel-wise, the Spearman rank correlation coefficient was calculated for evaluating the predicted priority ranking against the true priority rank. The ranking was ordered according to the cloud coverage calculated on each image. The predicted priority ranking used the output of NU-Net while the true priority rank used the image label of the test dataset.

$$\rho = 1 - \frac{\sum d_i^2}{n(n^2 - 1)}. \tag{5}$$

2.1.3. Cloud Coverage Calculation

The cloud coverage was calculated from the NU-Net segmentation result by evaluating the ratio of the cloud pixels and the total number of pixels of the image:

$$\text{Cloud Coverage} = \frac{\text{Number of predicted cloud pixels}}{\text{Total number of pixels}}. \quad (6)$$

This equation was implemented as follows:

$$\text{Cloud Coverage} = \frac{\sum_{i=1}^{a}\sum_{j=1}^{b} p_{i,j}}{(a)(b)}, \quad (7)$$

where a and b are the width and height of the image, respectively, and p the value of pixels that can be 0 or 1 as a result of the classification. In this way, the cloud coverage went from 0.0 when the image was fully clear to 1.0 when it was fully cloudy.

2.2. Cloud Coverage under Unseen Domain

CNN's cloud detection was also evaluated in the unseen domain. For this evaluation, the second dataset containing thumbnail images from FACSAT-1 was used. These images were not used during training and were acquired by a different sensor. While the training dataset contained Sentinel-2B images, this evaluation dataset contained CubeSat images that were obtained from a different imager. In addition, the size and resolution of this dataset were different from the training dataset. Table 3 summarizes the main differences of the two missions (Sentinel-2 and FACSAT).

Table 3. Summary of FACSAT-1 and Sentinel-2 missions.

Parameters	FACSAT-1	Sentinel-2
Spectral response	RGB	Multispectral (13 bands)
GSD	30 m	10, 20 and 60 m
Swath size	45 × 32 km	290 km
Orbit	Sun-synchronous at altitude 480 km	Sun-synchronous at altitude 786 km
Products used	Thumbnails image	True Color Image and Scene Classification map

The thumbnail images of the second dataset were resized to 64 × 48 pixels to match the NU-Net input. This process degraded the resolution, which is another challenge that must be overcome by a trained CNN. However, it is worth recalling that the images in the training dataset were also resized to 64 × 48 pixels as mentioned in Section 2.1.1.

Evaluation of the NU-Net with the second dataset was qualitative because these images did not have a corresponding label with a cloud mask. Qualitative evaluation was performed by visual comparison of the output of the NU-Net. All images in the dataset were visualized, and the same images were also visualized after ordering according to the cloud coverage.

2.3. Cloud Detection System on the COTS Embedded Device and Open-Source Framework

On-board cloud detection requires the implementation of the CNN on an embedded system. Even though most of the hardware applications of CNN are based on ASIC, GPU or FPGA, the NU-Net runs on a Microcontroller Unit (MCU) over micropython through the TinyML framework. MCU are cheaper, easier to acquire, and require a lower power than optimized hardware for CNN. Once the micropython was installed on the MCU, it could operate a reduced model of NU-Net which was obtained using TensorflowLite to convert the Keras model to the TinyML version. The selected COTS microcontroller was the ESP32, which is a 32 bits, 240 MHZ, 520 kB RAM single-core device. This was selected

because it supports the micropython version with the TinyML framework. The module was implemented and tested using the ESP32-CAM development board that includes an external RAM of 4 MB.

The proposed systems consists of an MCU running the NU-Net model, pre-trained to detect clouds in the image and programmed to calculate the cloud coverage from the image output of the NU-Net. The MCU works as a module that can receive an image and return the corresponding cloud coverage. Therefore, a master CPU sends the image and receives the cloud coverage to add this parameter and order the images accordingly. This CPU can be the main processor of the imager or the on-board computer of the small satellite. To work independently of the master, the MCU was programmed with a communication protocol through UART at 115,200 bps. Using this communication protocol, the master indicated that an image would be sent to the MCU, then a command to process the image was also sent, and finally the cloud coverage value was received Figure 2.

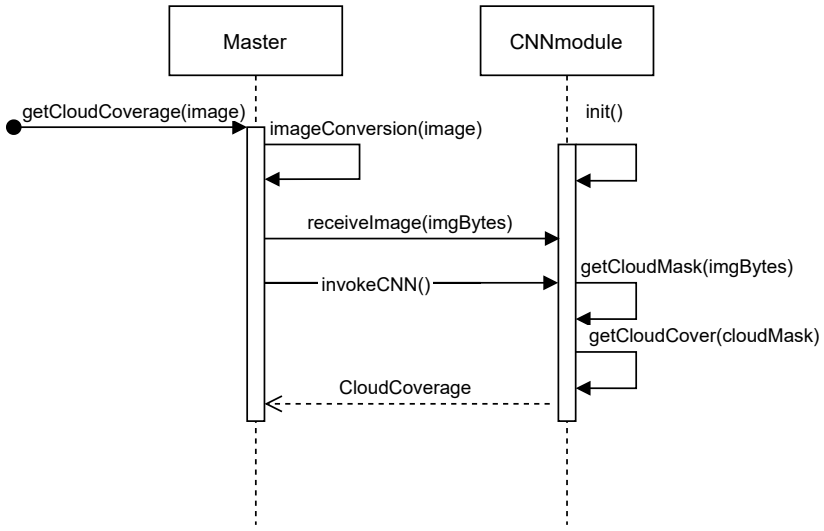


Figure 2. Information flow between the proposed module and a master device.

Performance of NU-Net under Unseen Domain

As explained above, NU-Net was trained using a dataset with Sentinel-2 images with a cloud mask used as a label. In addition, in this work, the performance of NU-Net was also evaluated under conditions given by a dataset from CubeSat images. There was no cloud mask for this dataset; for this reason, the metrics explained above could not be calculated. In this case, a qualitative evaluation was performed based on a visualization of the prioritization of the image. To evaluate the practicality of the proposed approach, the rank obtained by the embedded device was compared against the rank of four independent experts that operated the satellite and ranked the images during the CubeSat operation. In addition, we also compared the prioritization of NU-Net on a PC with the prioritization of NU-Net on MCU.

3. Results

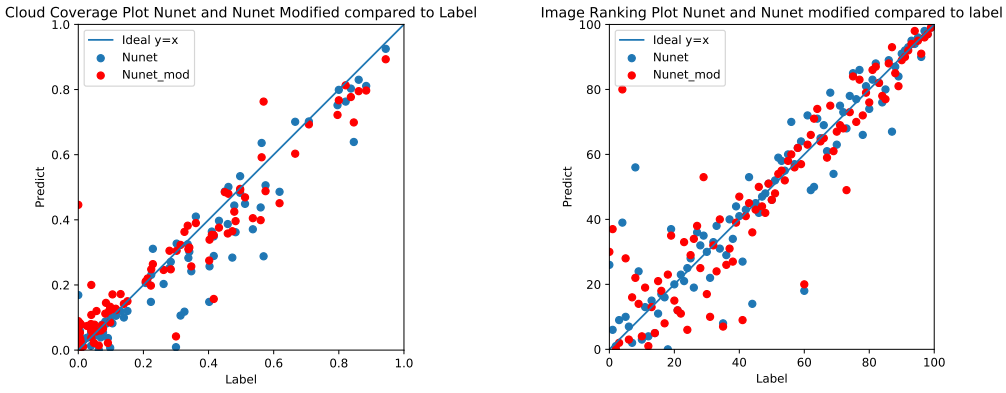
The cloud detection system was implemented according to the method described in the previous section. This section presents the results of the implementation of CNN on the embedded system that includes performance metrics and qualitative evaluation using visualization.

3.1. Cloud Detection and Image Prioritization

CNN was implemented on TensorFlow to evaluate the performance on a PC. Two versions of the CNN were trained and tested: NU-Net and its modified version, NU-Net-mod, whose architectures are described in the method section.

To evaluate the performance, CNN output was used to calculate cloud coverage, and these results were compared with the cloud coverage of labels. In the ideal expected case, both calculations must be equal; therefore, the visualization of predicted cloud coverage against the cloud coverage of level must be close to line $y = x$. This is visualized in Figure 3a for NU-Net and NU-Net-mod where most of the points follow the expected trend with a couple of outliers that will be analyzed in the last section.

The 100 images in the dataset were ranked according to cloud coverage. In the same way of comparing the cloud coverage of the label images with the predicted value by CNN, the ranking was also compared. The images of the test dataset were ranked using the cloud coverage of the label, the cloud coverage obtained from NU-Net, and the cloud coverage obtained from NU-Net-mod. The predicted rank versus the expected rank is shown in Figure 3b. The points follow the ideal case $y = x$ with some outliers.



(**a**) (**b**)

Figure 3. Comparison of predicted output by NU-Net and NU-Net-mod against the expected result of image labels. (**a**) cloud coverage; (**b**) rank.

Images of the test dataset were ordered according to the cloud coverage calculated from the labels (Figure 4). This ordering can be used for image prioritization; the image with the lowest value of cloud coverage is located in the top left position, and the image with the high cloud coverage is located on the bottom right. The order of the test dataset using the cloud coverage predictions of NU-Net and NU-Net-mod is shown in Figure 5a,b. Both Figures 4 and 5 show the ranking visualization results based on cloud coverage. Five images have been marked in each figure to help identify specific differences in the rank.

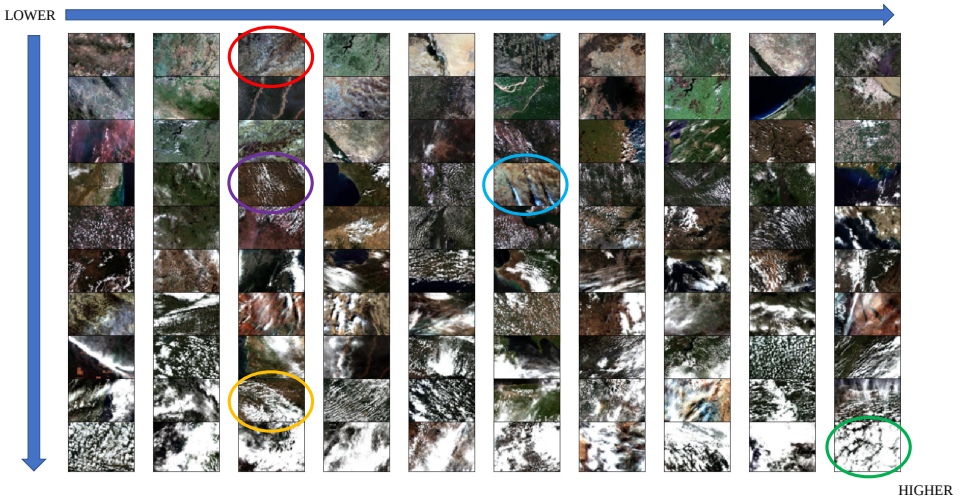
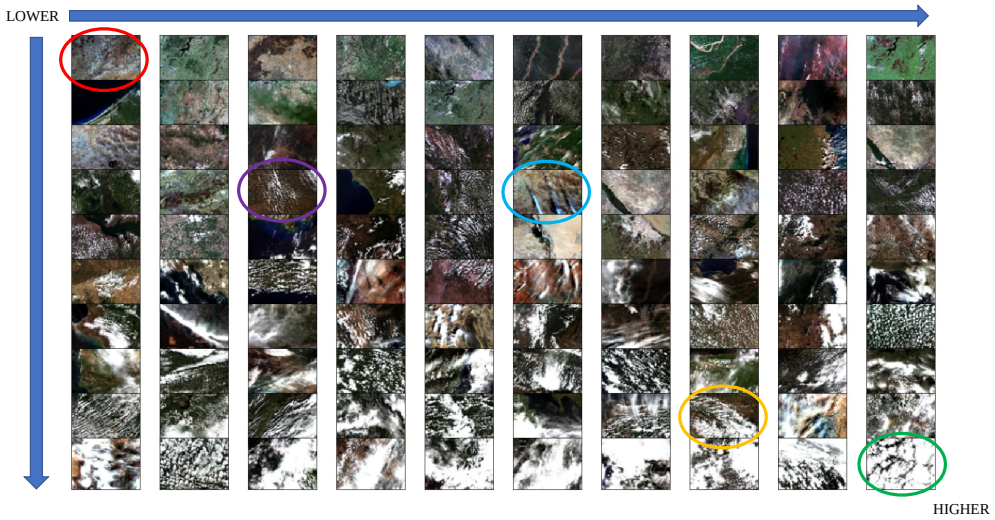
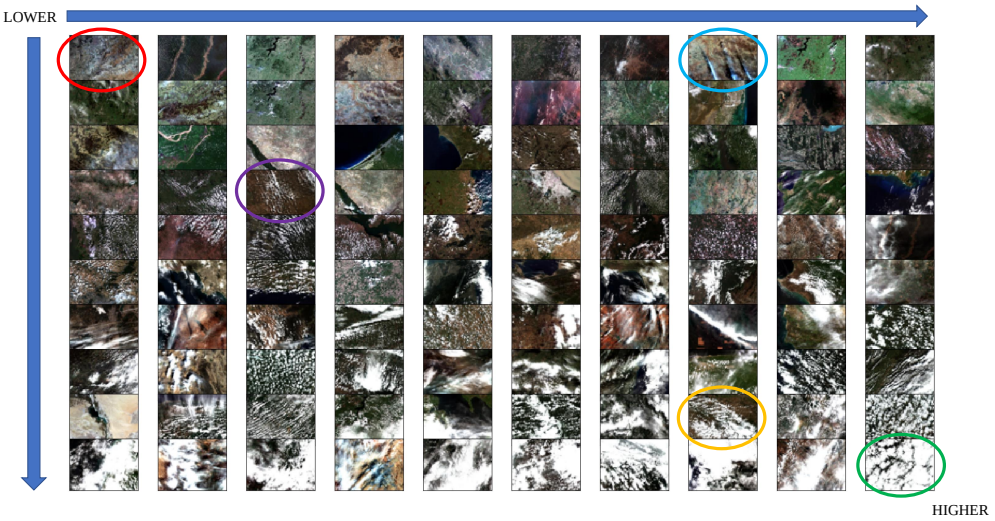


Figure 4. Rank of test images according to cloud coverage from labels. Colored circles are used to tag images.



(**a**)



(**b**)

Figure 5. Rank of test images according to cloud coverage obtained from CNN. Colored circles are used to tag images. (**a**) Results based on NU-Net; (**b**) Results based on NU-Net-mod.

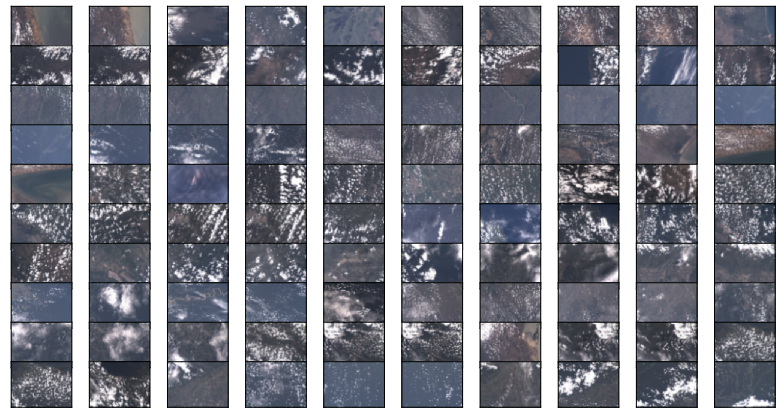
This confirms that both versions of CNN can be used to identify the cloud coverage of the images, likewise image prioritization. From Figure 3, it can be observed that the NU-Net-mod results are closer to the expected results with the exception of some outliers. However, it is necessary to calculate the metrics based on the confusion matrix to evaluate performance. Table 4 shows the comparison of the metrics for NU-Net and NU-Net-mod. Since NU-Net-mod shows a higher value on accuracy, recall, and F1, this architecture was used for generalization analysis.

Table 4. NU-Net and NU-Net-mod metric results.

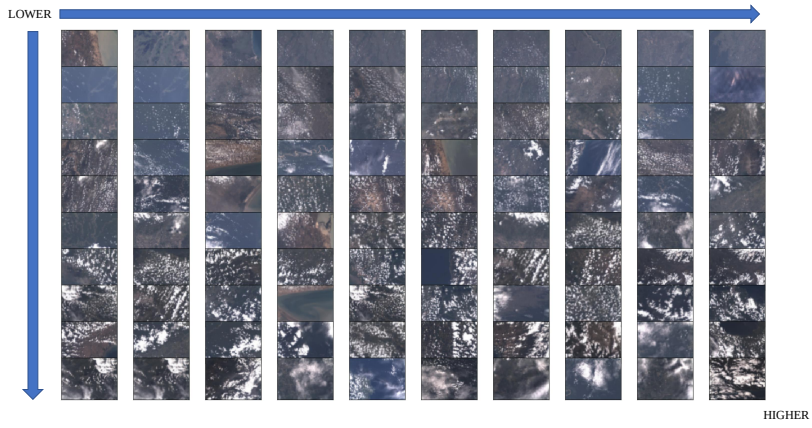
Metric	NU-Net	NU-Net-mod
Spearman Correlation	0.928	0.897
Accuracy	0.906	0.918
Recall	0.757	0.832
Precision	0.859	0.844
F1	0.804	0.838

3.2. Performance under Unseen Domain—CubeSat Dataset

To evaluate system performance when the test dataset comes from an unseen domain, 100 thumbnail images from FACSAT-1 that correspond to images of different imagers were used for testing. As explained above, there are no labels for this dataset; therefore, the evaluation was based on visual inspection as a human operator would do it. Figure 6a shows no specific order, while Figure 6b shows the ordered dataset according to cloud coverage. It can be observed that the order meets the requirement where images with high cloud coverage are located toward the bottom right. However, there are some images that are not correctly classified, and thus the rank is not correct, similar to the case of outliers identified with the previous dataset.



(**a**)



(**b**)

Figure 6. NU-Net-mod results under unseen domaing: images from FACSAT-1. (**a**) Sample of thumbnail images from FACSAT-1; (**b**) Rank thumbnail images from FACSAT-1.

3.3. Performance on Embedded Systems

Once the application of the CNN was demonstrated for both datasets, the embedded system was implemented as described in the method section to verify the performance over SoC that are not optimized for AI. This will allow the implementation of CNN on low-power devices that are suitable for nanosatellites. The evaluation was based on the quantitative method and visual inspection, and the two datasets were used during this evaluation.

The first evaluation consisted of a visual inspection of the result of one image to confirm that the cloud coverage was detected by the embedded system. In this case, the system was programmed to return the mask of the received image. Figure 7a shows the processed image and the comparison of the cloud mask obtained by the NU-Net-mod that runs on the PC and the NU-Net-mod that runs on the embedded system.

(**a**) (**b**) (**c**)

Figure 7. Comparison of cloud mask obtained from PC and from Embedded device. (**a**) Original Image; (**b**) Result obtained from NU-Net-mod in PC; (**c**) Result obtained from NU-Net-mod in Embedded device.

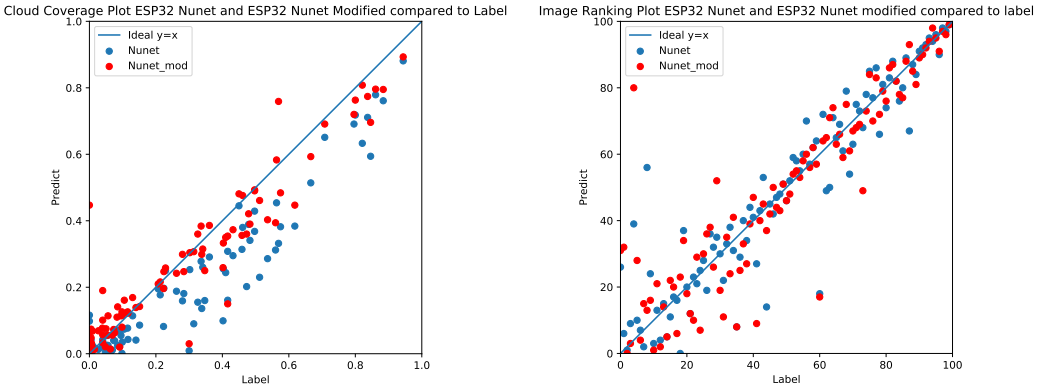
The time to obtain the cloud mask includes the transmission of the image, the processing by CNN on the embedded system, and the reception of the cloud mask through serial communication. This time was measured and is 6.1 s.

To evaluate the effect of quantization, NU-Net and NU-Net-mod were implemented on the embedded system, and they were tested for cloud detection using the Sentinel-2 dataset. Table 5 shows the metrics for NU-Net and NU-Net-mod when they run on the embedded system for 100 images of the test dataset.

Table 5. Metric results of NU-Net and NU-Net-mod on the embedded device.

Metric	NU-Net	NU-Net-mod
Spearman correlation	0.928	0.899
Accuracy	0.895	0.919
Recall	0.648	0.826
Precision	0.913	0.852
F1	0.758	0.839

Another result of the quantization evaluation is shown in Figure 8. NU-Net and NU-Net-mod in the embedded system were compared with the labels to identify which architecture was closest to the ideal case; a line following $y = x$. Cloud cover predictions based on NU-Net output are lower than NU-Net-mod predictions, and this is consistent with the metrics that indicated lower NU-Net performance. Therefore, the following results, which compare the performance of the CNN on the embedded system, are based on NU-Net-mod.

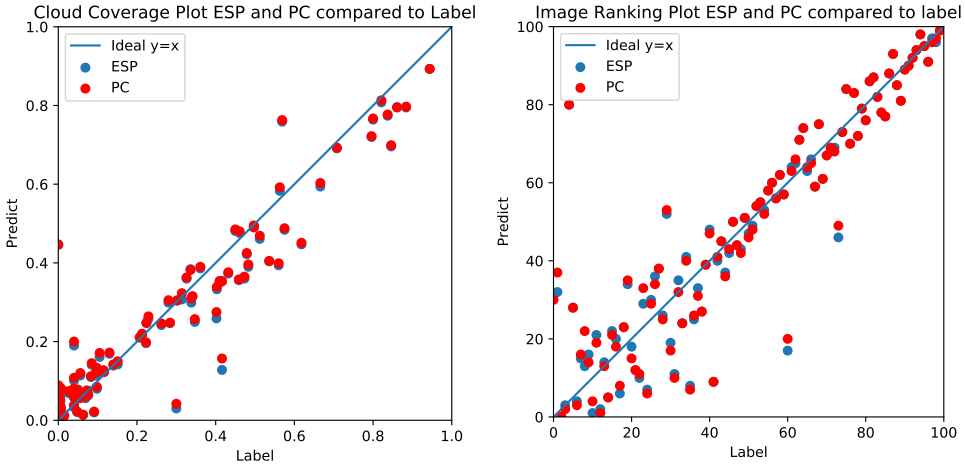


(**a**) (**b**)

Figure 8. Comparison of predicted output by NU-Net and NU-Net-mod against the expected result of image labels on embedded device ESP32. (**a**) Cloud coverage; (**b**) Rank.

3.3.1. Comparison of CNN Performance: Embedded System vs. PC

The embedded system was tested with NU-Net-mod and the 100 images of the test dataset. For NU-Net-mod predictions, the cloud coverage was calculated and compared with the cloud coverage of the labels. The same procedure was applied with the implementation of NU-Net-mod on PC, and all results are shown in Figure 9a. The 100 images were ranked according to cloud coverage; however, there are three possible ranks: (1) rank using cloud coverage of the labels; (2) rank using embedded system results; and (3) rank using PC results. The ideal case is that the ranking based on the cloud coverage predictions is closer to the ranking of the labels. Figure 9b shows the rank of the label versus the rank based on embedded systems and the rank of the label versus the rank based on PC. The predictions based on the embedded systems are similar to the predictions on the PC, and there is a slight difference due to the quantization.



(**a**) (**b**)

Figure 9. Comparison of predicted output by NU-Net-mod on PC vs. embedded device. (**a**) Cloud coverage; (**b**) Rank.

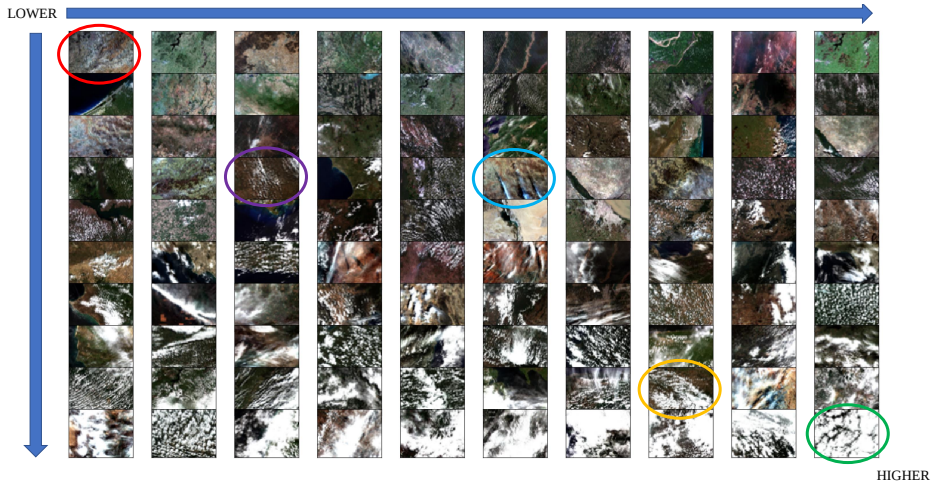
As a qualitative evaluation, a comparison of the PC rank and the embedded system rank is presented in Figure 10. The difference can be observed mainly in the first three rows; these images have low cloud coverage. For a quantitative evaluation, the metrics are presented in Table 6.

Table 6. Comparison of metrics: PC vs. Embedded device.

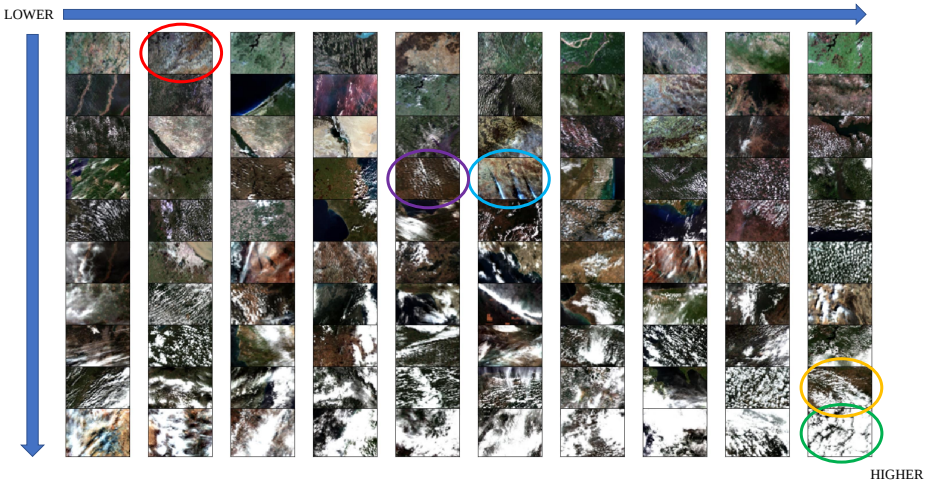
Metric	PC	Embedded Device
Spearman correlation	0.897	0.899
Accuracy	0.918	0.919
Recall	0.832	0.826
Precision	0.844	0.852
F1	0.838	0.839

3.3.2. Comparison of CNN Performance: Embedded System under Unseen Domain—CubeSat Dataset

The performance of the embedded system against the PC under an unseen domain was also evaluated using the CubeSat dataset. The cloud coverage of 100 images of the FACSAT-1 dataset was calculated using NU-Net-mod on the embedded system and on the PC. Figure 11 shows the rank obtained by the embedded systems and the rank obtained by the PC. It can be seen that there are some differences between the ranks. However, most of the images have low cloud coverage. The quantization effect is more observable when cloud coverage is similar among several images, which affects the ranking.



(**a**)

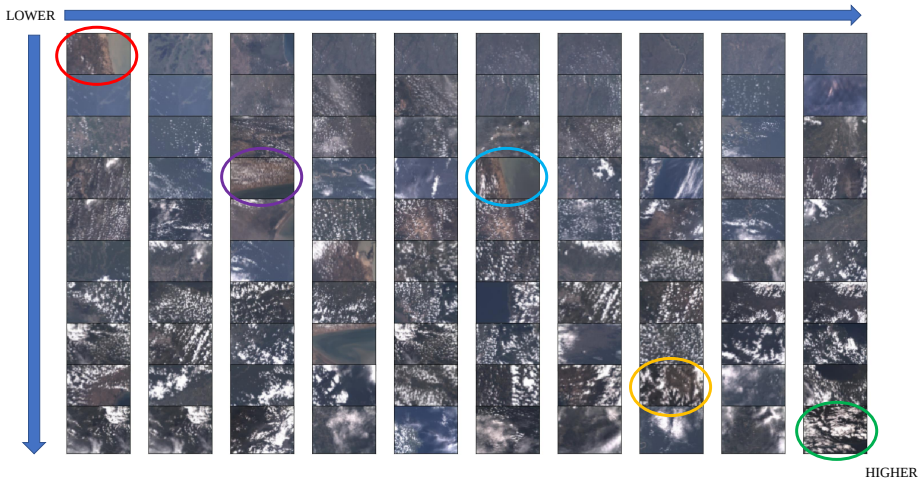


(**b**)

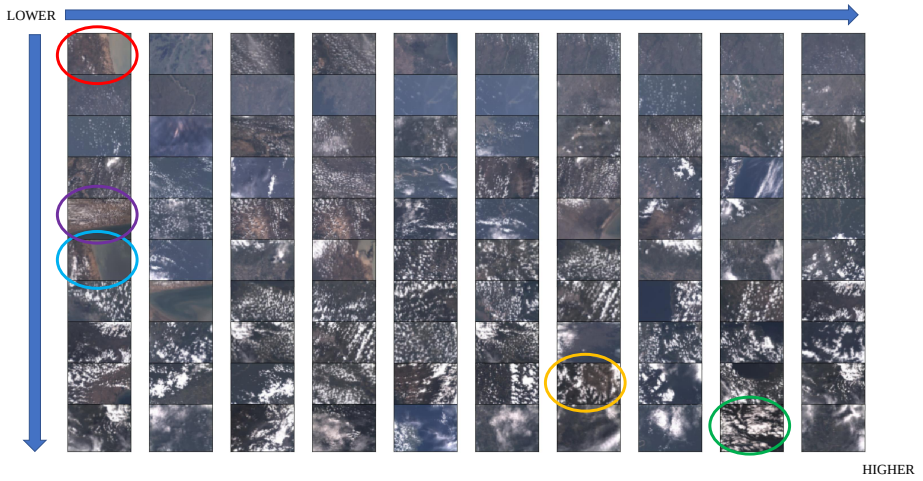
Figure 10. Comparison of PC rank vs. Embedded device rank in Sentinel-2 images. Colored circles are used to tag images. (**a**) Rank by PC; (**b**) Rank by embedded device.

To highlight the difference between the embedded system rank and the PC rank, the cloud coverage and the rank were plotted to see how close they were to the ideal case, $y = x$ (Figure 12). There are no significant outliers, and most of the values are close to the ideal case. This is true for both cloud coverage and ranking (Figure 12a,b), respectively. It can be seen that the calculated maximum cloud coverage is around 0.4, which is consistent with the low cloud coverage visualized in the previous comparison.

Another way of evaluating the applicability of the proposed approach of using the trained CNN with Sentinel-2 images to rank the CubeSat images according to cloud coverage consists of comparing it against the rank developed by the expert operators of the satellite. Taking into account, subjectively, the human criteria, four satellite operators were asked to rank five randomly selected images to emulate a daily routine. The comparison of the four ranks against the CNN ranking in the embedded device is shown in Figure 13. It can be observed that the embedded device and human operator agree to select images with lower cloud coverage. Regarding the higher cloud coverage, there are differences because operators are trained to count the gaps in the clouds when estimating cloud coverage, while CNN is trained to segment the clouds ignoring the gaps.

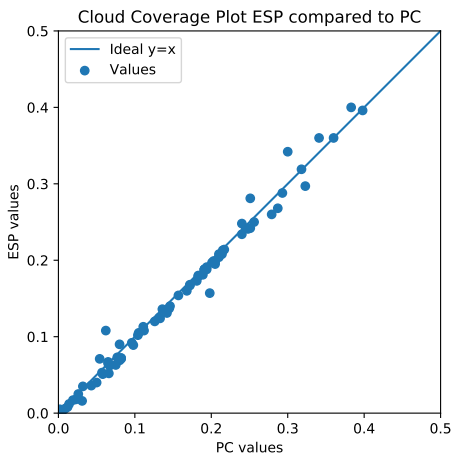


(**a**)

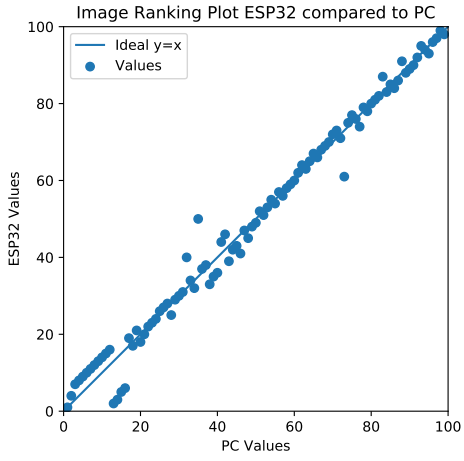


(**b**)

Figure 11. Comparison of PC rank vs. embedded device rank in FACSAT-1 images. Colored circles are used to tag images. (**a**) Rank by PC; (**b**) Rank by embedded device.



(**a**)



(**b**)

Figure 12. Comparison of PC results vs. embedded device results on FACSAT-1 images. (**a**) Cloud coverage: PC vs. embedded device; (**b**) Rank: PC vs. embedded device.

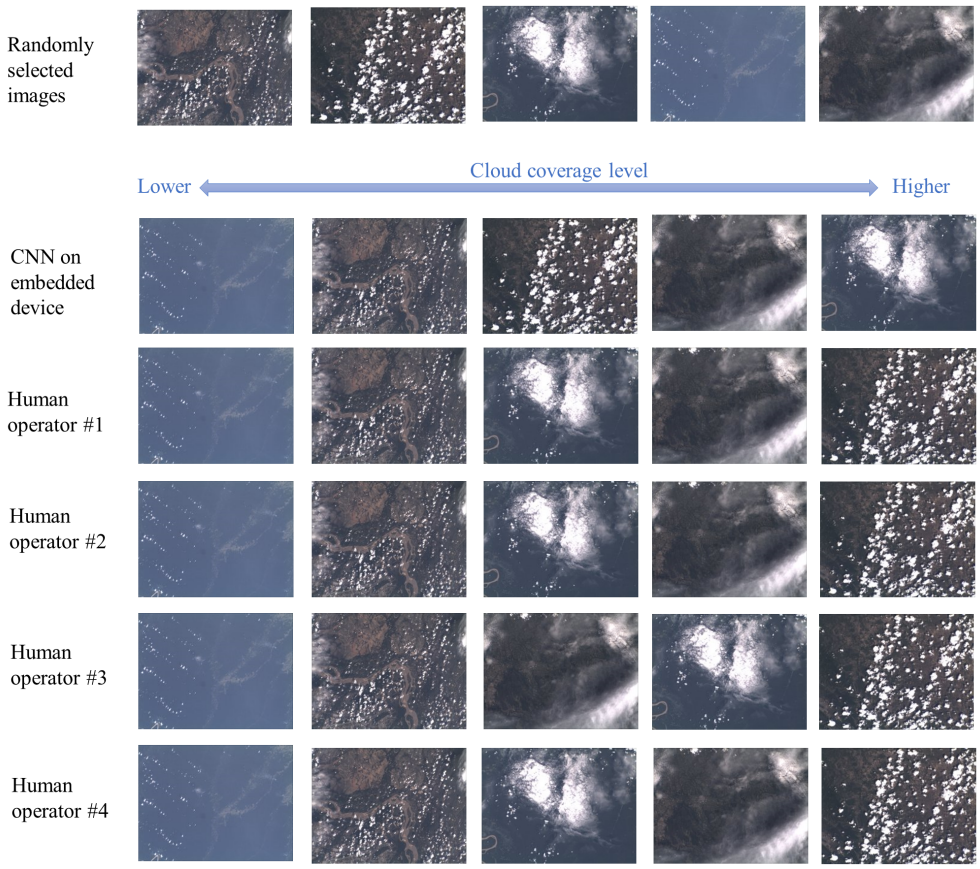


Figure 13. Rank comparison among CNN on embedded device and four human operators.

4. Discussion

The results have shown that both the NU-Net architecture and the variant NU-Net mod are able to detect cloud coverage on thumbnail images. The performance metric indicated an accuracy of more than 0.9 for both architectures; however, the NU-Net-mod performed better on recall, as shown in Table 4 in the Results section. A key result of NU-Net-mod is the ability to detect cloud coverage in unseen domain images. When NU-Net-mod is tested with images of a different sensor, the images of FACSAT-1 Cubesat, it can be seen that the cloud coverage was identified and the images in this dataset were ranked according to the cloud level (Figure 6).

In addition, we demonstrate the good performance of NU-Net-mod on the unseen domain. Another key result of this work is the demonstration of its implementation in an embedded system using COTS components. This result encourages the implementation of CNN on-board small satellites such as Cubesat, nanosatellites, and picosatellites, where power constraints limit the use of AI-optimized processors. The results of Section 3.3 demonstrated that the trained CNN could be implemented in embedded systems with little effect on performance. The prediction of the embedded system and the PC prediction are practically the same as shown in the comparison of Figure 9.

The NU-Net and NU-Net-mod had similar results for both datasets, and both architectures were implemented in the embedded system. However, NU-Net-mod was used for most tests and comparisons against PC results because NU-Net-mod showed slightly better results according to the metrics. However, comparing the prediction of NU-Net-mod with the expected case reveals some outliers (Figure 3), which can cause errors during image prioritization.

The outliers identified and shown in Figure 14 correspond to the five figures with the greatest error in calculating the cloud coverage. These images are part of the test

dataset that comprises 100 images. The images are identified by the following identification numbers: #25, #76, #79, #83, #94.

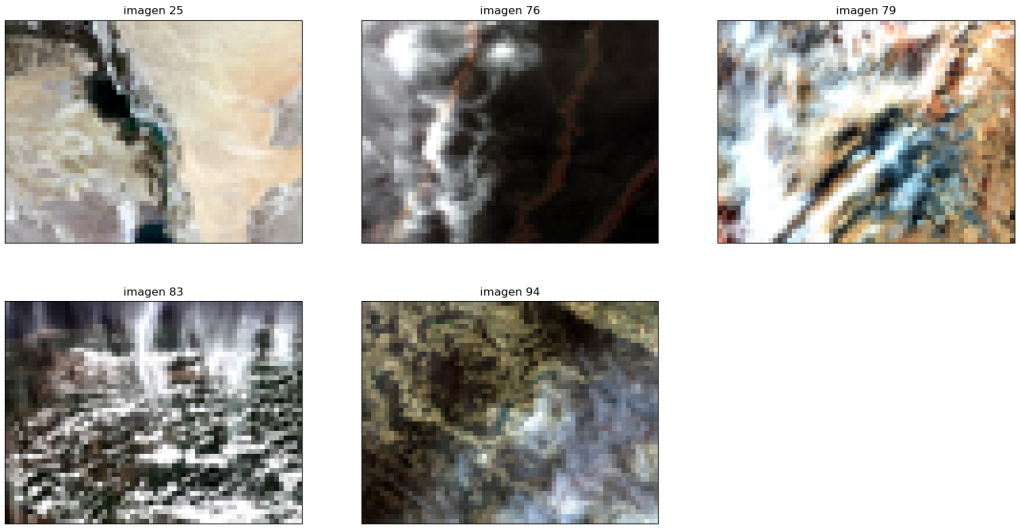


Figure 14. Image with highest errors on cloud coverage by NU-Net-mod.

The label and the cloud mask predicted by the NU-Net-mod on image #25 are shown in Figure 15. According to the label cloudless image, the expected cloud coverage is 0.0; however, the calculated cloud coverage was 0.447 and it can be seen that a big area was wrongly classified as cloud. This image had been analyzed by Park [24] and identified as a bright scene, which presented problems for several methods. It is important to highlight that this image is an outlier for NU-Net-mod but it was correctly evaluated for NU-Net. A similar case is noted with the image #79, whose expected cloud coverage is 0.569 and the calculated value is 0.759. For image #79, the label and the predicted cloud mask are shown in Figure 16. To improve the performance on bright scenes such as clouds above snow/ice surfaces, thermal bands and geographical or seasonal information must be considered [24].

(**a**) (**b**) (**c**)

Figure 15. Comparison Predict vs. Label image 25. (**a**) Image 25; (**b**) output for image 25; (**c**) Image label 5.

(**a**) (**b**) (**c**)

Figure 16. Comparison Predict vs. Label image 79. (**a**) Image 79; (**b**) Output for image 79; (**c**) Label for image 79.

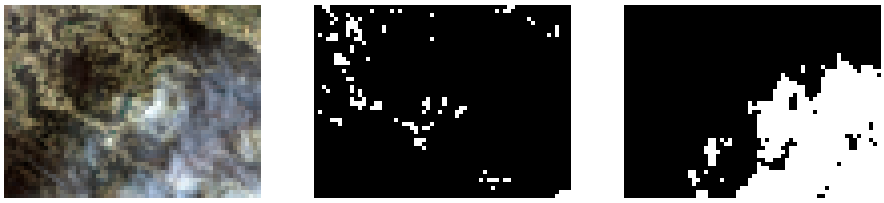
The images #76, #83 and #94 are shown with the corresponding labels and predicted cloud mask in Figures 17–19, whose expected cloud coverages are 0.416, 0.618 and 0.300, respectively, and the predicted values are 0.15, 0.447 and 0.03. In these images, the cirrus clouds are wrongly classified. For example, the prediction in image #83 is better than the label. This is caused by the inconsistency in the dataset where some cirrus clouds are marked as cloud whereas they do not appear in other images, as explained previously by Park [24]. If cirrus detection is critical for the application, the proposed approach can be extended by including additional bands at the cost of increasing the computational burden.

(**a**) (**b**) (**c**)

Figure 17. Comparison Predict vs. Label image 76. (**a**) Image 76; (**b**) Output image 76; (**c**) Label image 76.

(**a**) (**b**) (**c**)

Figure 18. Comparison Predict vs. Label image 83. (**a**) Image 83; (**b**) Output image 83; (**c**) Label image 83.



(**a**) (**b**) (**c**)

Figure 19. Comparison Predict vs. Label image 94. (**a**) Image 94; (**b**) Output image 94; (**c**) Label image 94.

5. Conclusions

We proposed an autonomous system to quantify the cloud coverage in RGB images that is based on Machine Learning and is implemented on a COTS microcontroller, which is suitable to be used on board small satellites, CubeSats, and pico/nano/microsatellites. This system receives the image through a serial interface and returns the cloud coverage index after the evaluation. Because of this interface, the system can be connected directly to the imager or to the on-board computer to be used for image prioritization by selecting the images with the lowest cloud coverage to be downloaded.

The proposed system uses a Convolutional Neural Network (CNN) based on the architecture NU-Net, which has been optimized to run on a microcontroller. In this work, the NU-Net architecture and a variant NU-Net-mod were trained using a dataset from Sentinel-2 and a quantized version of CNN was run on a general-purpose COTS microcontroller

using TinyML. Both architectures have shown an accuracy greater than 0.9. NU-Net-mod showed slightly better results in recall and accuracy and was less affected by quantization. However, it presented greater errors when tested with bright scenes. NU-Net performed better for this kind of image.

The trained CNN was tested under an unseen domain using a dataset from a different imager. In addition to the Sentinel-2 images that were used for training and testing, a second dataset was also used for the evaluation of NU-Net and NU-Net-mod. The images downloaded from CubeSat were processed by the CNN, and the cloud coverage was quantified. The results demonstrated that the architecture can generalize to an unseen domain of images.

This research demonstrated that cloud coverage estimation can be achieved on a general-purpose microcontroller using an open-source framework without specific preprocessing of the images. Using a serial interface, the proposed systems can be integrated on a CubeSat to quantify the cloud coverage of the images. In this task, autonomous systems can help to save time during satellite operations because images are evaluated prior to download; this way, satellite operators can first request images with low cloud coverage.

Author Contributions: Conceptualization, J.G.-L., C.S., L.C. and J.M.; methodology, J.G.-L. and C.S.; software, C.S.; validation, J.G.-L. and C.S; formal analysis, J.G.-L., C.S.; investigation, J.G.-L., C.S., L.C. and J.M.; resources, J.G.-L., C.S. and L.C.; data curation, L.C.; writing—original draft preparation, J.G.-L. and C.S.; writing—review and editing, J.G.-L., C.S., L.C., J.M, S.R., J.R.-F. and I.F.A.; visualization, J.G.-L. and C.S.; supervision, S.R.; project administration, S.R., J.R.-F. and I.F.A.; funding acquisition, S.R., J.R.-F. and I.F.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Ministerio de Ciencia, Tecnología e Innovación de Colombia (MINCIENCIAS) and the Colombian Air Force (COLAF).

Data Availability Statement: The data used in this research are from the publicly archived dataset available at http://www.mdpi.com/2072-4292/12/23/3941/s1, accessed on 1 November 2022 and thumbnail images from FACSAT-1 mission.

Acknowledgments: The authors would like to thank the LEOpar mission development team. The authors are also thankful to Sergio Barrera and all operators of FACSAT-1 for collaborating in the evaluation of cloud cover of FACSAT-1 images.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial intelligence
ASIC	Application-specific Integrated Circuit
COTS	Commercial off-the-shelf
CNN	Convolutional Neural Network
CPU	Central processing unit
FN	False negative
FP	False positive
FPGA	Field programmable gate arrays
GPU	Graphics Processing Unit
ML	machine learning
MCU	Microcontroller Unit
PC	Personal computer
RGB	Red Green Blue
TN	True negative
TP	True positive
VPU	Vision Processing Unit

References

1. Sweeting, M.N. Modern Small Satellites-Changing the Economics of Space. *Proc. IEEE* **2018**, *106*, 343–361. [CrossRef]
2. Crusan, J.; Galica, C. NASA's CubeSat Launch Initiative: Enabling broad access to space. *Acta Astronaut.* **2019**, *157*, 51–60. [CrossRef]
3. Villela, T.; Costa, C.A.; Brandão, A.M.; Bueno, F.T.; Leonardi, R. Towards the thousandth CubeSat: A statistical overview. *Int. J. Aerosp. Eng.* **2019**, *2019*, 5063145. doi: [CrossRef]
4. Gonzalez-Llorente, J.; Lidtke, A.A.; Hatanaka, K.; Limam, L.; Fajardo, I.; Okuyama, K.I. In-orbit feasibility demonstration of supercapacitors for space applications. *Acta Astronaut.* **2020**, *174*, 294–305. [CrossRef]
5. Poghosyan, A.; Golkar, A. CubeSat evolution: Analyzing CubeSat capabilities for conducting science missions. *Prog. Aerosp. Sci.* **2017**, *88*, 59–83. [CrossRef]
6. Spence, H.E.; Caspi, A.; Bahcivan, H.; Nieves-Chinchilla, J.; Crowley, G.; Cutler, J.; Fish, C.; Jackson, D.; Jorgensen, T.M.; Klumpar, D.; et al. Achievements and Lessons Learned From Successful Small Satellite Missions for Space Weather-Oriented Research. *Space Weather* **2022**, *20*, e2021SW003031. [CrossRef]
7. Mitry, M. Routers in space: Kepler communications' CubeSats will create an Internet for other satellites. *IEEE Spectr.* **2020**, *57*, 38–43. [CrossRef]
8. Selva, D.; Krejci, D. A survey and assessment of the capabilities of Cubesats for Earth observation. *Acta Astronaut.* **2012**, *74*, 50–68. [CrossRef]
9. Sandau, R.; Roeser, H.P.; Valenzuela, A.; Borgeaud, M.; Scheidegger, N.; Noca, M.; Roethlisberger, G.; Jordan, F.; Choueiri, T.; Steiner, N. *Small Satellite Missions for Earth Observation*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 207–213. [CrossRef]
10. Nagel, G.W.; Novo, E.; Kampel, M. Nanosatellites applied to optical earth observation: A review. *Rev. Ambiente Agua* **2020**, *15*, 1–19. [CrossRef]
11. Altena, B.; Mousivand, A.; Mascaro, J.; Kääb, A. Potential and limitations of photometric reconstruction through a flock of dove cubesats. In Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences—ISPRS Archives, Jyväskylä, Finland, 25–27 October 2017; Volume 42, pp. 7–11. [CrossRef]
12. Giovanni, C.; Eliot, B. Technology transfer and capability building for Colombia's space program by means of small satellites. In Proceedings of the International Astronautical Congress, IAC, Washington, DC, USA, 21–25 October 2019; Volume 2019.
13. Furano, G.; Meoni, G.; Dunne, A.; Moloney, D.; Ferlet-Cavrois, V.; Tavoularis, A.; Byrne, J.; Buckley, L.; Psarakis, M.; Voss, K.O.; et al. Towards the Use of Artificial Intelligence on the Edge in Space Systems: Challenges and Opportunities. *IEEE Aerosp. Electron. Syst. Mag.* **2020**, *35*, 44–56. [CrossRef]
14. Ajani, T.S.; Imoize, A.L.; Atayero, A.A. An Overview of Machine Learning within Embedded and Mobile Devices–Optimizations and Applications. *Sensors* **2021**, *21*, 4412. [CrossRef] [PubMed]
15. Drönner, J.; Korfhage, N.; Egli, S.; Mühling, M.; Thies, B.; Bendix, J.; Freisleben, B.; Seeger, B. Fast Cloud Segmentation Using Convolutional Neural Networks. *Remote Sens.* **2018**, *10*, 1782. [CrossRef]
16. Wang, X.; Han, Y.; Leung, V.C.M.; Niyato, D.; Yan, X.; Chen, X. Convergence of Edge Computing and Deep Learning: A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 869–904. [CrossRef]
17. Yao, Y.; Jiang, Z.; Zhang, H.; Zhou, Y. On-Board Ship Detection in Micro-Nano Satellite Based on Deep Learning and COTS Component. *Remote Sens.* **2019**, *11*, 762. [CrossRef]
18. Maskey, A.; Cho, M. CubeSatNet: Ultralight Convolutional Neural Network designed for on-orbit binary image classification on a 1U CubeSat. *Eng. Appl. Artif. Intell.* **2020**, *96*, 103952. [CrossRef]
19. Azami, M.H.b.; Orger, N.C.; Schulz, V.H.; Oshiro, T.; Cho, M. Earth Observation Mission of a 6U CubeSat with a 5-Meter Resolution for Wildfire Image Classification Using Convolution Neural Network Approach. *Remote Sens.* **2022**, *14*, 1874. [CrossRef]
20. Giuffrida, G.; Diana, L.; de Gioia, F.; Benelli, G.; Meoni, G.; Donati, M.; Fanucci, L. CloudScout: A Deep Neural Network for On-Board Cloud Detection on Hyperspectral Images. *Remote Sens.* **2020**, *12*, 2205. [CrossRef]
21. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *LNCS Lect. Notes Comput. Sci.* **2015**, *9351*, 234–241. [CrossRef]
22. Zhaoxiang, Z.; Iwasaki, A.; Xu, G.; Song, J. Cloud detection on small satellites based on lightweight U-net and image compression. *J. Appl. Remote. Sens.* **2019**, *13*, 1. [CrossRef]
23. Jeppesen, J.H.; Jacobsen, R.H.; Inceoglu, F.; Toftegaard, T.S. A cloud detection algorithm for satellite imagery based on deep learning. *Remote. Sens. Environ.* **2019**, *229*, 247–259. [CrossRef]
24. Park, J.H.; Inamori, T.; Hamaguchi, R.; Otsuki, K.; Kim, J.E.; Yamaoka, K. Rgb image prioritization using convolutional neural network on a microprocessor for nanosatellites. *Remote Sens.* **2020**, *12*, 3941. [CrossRef]
25. Bäuerle, A.; van Onzenoodt, C.; Ropinski, T. Net2Vis—A Visual Grammar for Automatically Generating Publication-Tailored CNN Architecture Visualizations. *IEEE Trans. Vis. Comput. Graph.* **2021**, *27*, 2980–2991. [CrossRef] [PubMed]