EUROPEAN WORKSHOP ON ON-BOARD DATA PROCESING (OBDP2021), 14-17 JUNE 2021

IN SPACE IMAGE PROCESSING USING AI EMBEDDED ON SYSTEM ON MODULE: EXAMPLE OF OPS-SAT CLOUD SEGMENTATION

Frédéric Férésin[1,2], Erwann Kervennic[1], Yves Bobichon[1,2], Edgar Lemaire[3,8], Nassim Abderrahmane[1], Gaétan Bahl[1,6], Ingrid Grenet[1,7], Matthieu Moretti[5], and Michael Benguigui[1,4]

[1]*IRT Saint Exupery, France (name.lastname@irt-saintexupery.com)*
[2]*Thales Alenia Space, France (name.lastname@thalesaleniaspace.com)*
[3]*Thales Research Technology, France (name.lastname@thalesgroup.com)*
[4]*ActiveEon, France (name.lastname@activeeon.com)*
[5]*Elsys Design, France (name.lastname@elsys-design.com)*
[6]*Inria, France (name.lastname@inria.fr)*
[7]*MyDataModels, France (name.lastname@mydatamodels.com)*
[8]*Univ. Côte d'Azur / LEAT / CNRS (name.lastname@unice.fr)*

ABSTRACT

This paper presents the in-flight results of different Artificial Neural Network and an evolutionary algorithm as challenger, performing the inference of the large (4M-pixels) images on-board OPS-SAT (ESA Cubesat demonstrator). Artificial Intelligence is a disruptive technology to process the sensors data directly on-board satellites. This technology is used to extract then transmit only the useful data to the end users, reducing the needs for data transmission. Artificial intelligence consequently changes the space observation paradigm, allowing an alert in real time from space through a low data rate downlink. This use case of cloud segmentation on 4M-pixels images demonstrates that on-board processing in close real time is already feasible with Deep Learning algorithms deployed on low-power chip, such as the System On Chip integrated on this ESA Cubesat.

Key words: Artificial Intelligence; System on Chip; FPGA; nano satellite; space; cloud segmentation; embedded systems.

1. INTRODUCTION

On the 18th December 2019, the OPS-SAT[1] CubeSat from ESA was launched with tens of experiments aboard, among which the Artificial Neural Networks (ANNs) developed by our team from IRT Saint Exupery on the Cyclone-V FPGA. Early 2021, ESA run our experiment of which the results are presented in this paper.

[1]OPS-SAT CubeSat: `https://directory.eoportal.org/web/eoportal/satellite-missions/o/ops-sat`

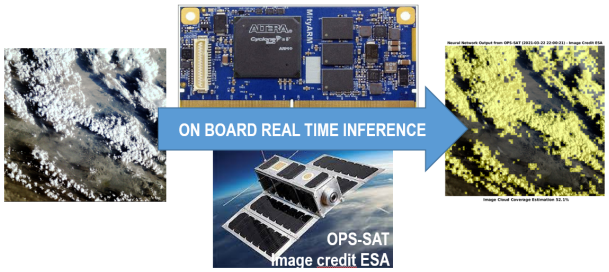

Figure 1: 22nd March 2021: the first image inferred on-board by an Artificial Neural Network running on OPS-SAT System On Chip (Cyclone V)

As explained in [1], considering the OPS-SAT camera performances, and the ground sampling distance of 53m per pixel at 600km altitude, the detection of clouds was selected, as a useful service to download only the not cloudy images. The difficulty to distinguish cloud and snow with RGB spectral bands is obvious, but the main objective was to demonstrate the feasibility to perform in-flight real-time inference by Artificial Neural Networks. Once this demonstration done, the next major step was to evaluate the processing performance on several images stored on-board.
First, to take full advantage of this ESA demonstrator opportunity, we developed different ANN architectures, described in Section 2. To challenge these neural networks we also implemented, on the Hard Processing System (HPS), a model obtained from a genetic programming algorithm called ZGP (see 2.4). Section 3 explains the method applied to generate the training dataset as representative as possible of the diversity of cloud shapes and colors, with few OPS-SAT images available. The on-board data test is also detailed.
Our last development step, detailed in Section 4, was to

deploy these ANNs on a the FPGA (Field Programmable Gate Array) part of a space-grade Cyclone-V System On Module. Section 5 provides and analyses the metrics of the inferences performed on-board OPS-SAT. Section 6 identifies the lessons learned and the resulting future improvements.

2. AI ALGORITHMS IMPLEMENTED ON-BOARD OPS-SAT

The first challenge consisted in selecting the Neural Networks architectures compliant with the limited capability of the execution board and the limited training data set. In the literature, neural networks have already been designed to detect and remove clouds together with their shadow [2], but require more processing power than the one available in OPS-SAT FPGA. Table 1 summarizes the four ANN architectures we selected corresponding to the experiments implemented on OPS-SAT and described in the next subsections.

FCN	=	Fully Convolutional Network (LeNet_FCN)
	=	C3.5.1,R,P2,C5.5.1,R,P2,C1.1.1,SG
	$\rightarrow$	nb of params: 614
CNN	=	Convolutional Neural Network
	=	C5.3.1,R,P2,C5.5.1,R,P2,D10,R,D2,SM
	$\rightarrow$	nb of params: 1429
HNN	=	Hybrid spiking Neural Network
	=	C5.3.1,R,P2,C5.5.1,R,P2,hSG,D10,R,D2,SM
	$\rightarrow$	nb of params: 1688
SNN	=	fully Spiking Neural Network
	=	C6.3.1,R,C6.3.1,R,D10,R,D2,SM
	$\rightarrow$	nb of params: 2666
C$k.n.d$	=	Conv2D(k filters, $n \times n$, dilation rate=d)
Pn	=	MaxPooling($n \times n$)
Dn	=	Dense(n neurons)
R	=	Relu
SM	=	SoftMax
(h)SG	=	(hard)Sigmoid

Table 1: Neural network architectures

2.1. CNN hardware architectures

A preliminary assessment of implementation on Cyclone V FPGA implies to select a very small convolutional architecture as solution, inspired by LeNet-5, with two convolutions (3 and 5 filters) and two fully-connected layers (10 and 2 neurons). To avoid implementing the detection task we applied classification on very small patches of 28x28 pixels. This size allowed pre-validating our HDL architecture with the MNIST dataset. Thus we classify very small areas (784 pixels) on the full image (about 4M pixels) as "cloud" or "no cloud" patches, considering that large clouds cover anyway hundreds of pixels.

2.2. Spiking Neural Network

Spiking Neural Networks (SNNs) [3] are often considered as the third generation of Neural Networks. Deriving from neurosciences, they mimick the event-based behaviour of the neural networks found in the brain. Rather than real-values, spiking neurons encode information into spikes, which are modeled by 1-bit signals. Therefore, hardware implementations of SNNs get rid of the costly Multiplication-Accumulation (MAC) operation involved in the activation integration process : when an input spike is received, its corresponding synaptic weight is simply accumulated (AC). When the accumulator overpasses a given threshold, an output spike is emitted. Using this binary event-based paradigm, SNNs try to approach the computational efficiency of the biological brain, drastically reducing logic resources required for their implementations [4].

2.2.1. Hybrid architecture

The Hybrid NN uses both a conventional Formal convolution stage and a spiking fully-connected stage. The aim of this novel hybrid architecture is to draw advantages from the disparity of spikes in the topology during inference. As shown in [3], the highest spiking density occurs in the first layers, which can counterbalance the SNN advantages in some particular layers.
The feature extraction stage (Convolution and Pooling) is the same as in Section 2.1. Its output feature maps are stored in FIFOs. A Transcoding Module translates pixels into spike trains according to rate-coding policy, and transmits those events to the spiking dense layers. The spiking dense layers are implemented in a fully-parallel fashion, i.e. each neuron of the model is physically implemented by an Integrate & Fire neuron [5]. A Terminate Delta module is used to detect the most active output neuron, thus enacting the winning class. The results of the implementation in terms of logic resources, power consumption estimation, latency and accuracy are presented in Table 2.

2.2.2. Full spiking architecture

In order to fully exploit the advantages brought by SNNs, we propose an architecture in which all the layers of the neural network are implemented in the spiking domain. This fully spiking architecture comprises a 'transcoding' and a 'classification' modules. These spike generator (transcoding) and classification modules, representing the input and the output of the SNN, are connected to a series of convolutional and fully-connected layers, represented by dedicated NPU (Neural Processing Unit) modules. These NPU modules are implemented in a way that guarantees an event-based communication protocol fitting the spiking data dynamics. The architecture uses mostly time-multiplexed computing which allows for reducing the hardware usage. A more detailed description of the architecture is available in [6].

2.3. Fully Convolutional Network

We then focused on a different approach, expected as more efficient with respect to our cloud detection task: the semantic segmentation. While CNN and SNN convert a 28x28 RGB patch into a single-pixel segmentation map, FCN converts that patch into a more precise 4x4-pixels segmentation map. Precise semantic segmentation tasks are usually solved using skip-connections between the encoding and decoding layers, as in UNet[7]. These skip-connections are useful to send high-frequency information to the decoder part of the network, while the encoder part tends to act like a low-pass filter because of successive convolutions. Clouds getting few high-frequency information, to remove the skip-connections does not decrease the accuracy of the neural network, as shown in Table 3 of [8], whereas the Table 4 shows that removing these skip-connections saves a significant amount of memory during network inference.
In Table 1, we thus present a tiny but competitive Fully Convolutionnal Network (FCN) that reaches an IoU of 74.16% with 614 parameters on the 38-Cloud [9] dataset where U-Net reaches 86.08% with around 31M parameters, and where f-Mask [10] reaches 75.16%.

2.4. Zoetrope Genetic Programming

In order to challenge neural networks in the cloud detection task, we also implemented a model resulting from an original genetic programming approach for symbolic regression called Zoetrope Genetic Programming (ZGP) [11]. Briefly, ZGP generates and evolves individuals that are mathematical expressions combining input variables, for instance the pixel RGB components. Its particularity lays in the way these individuals are genetically-optimised, which avoids overgrown and too complex models. After several generations where individuals are mutated and recombined to fit the data, the "fittest" individual is returned as the final model.
ZGP has several advantages among which its frugality regarding training data, the low training and inference computational time, the interpretability of the models and their easy implementation and portability on embedded systems.
In this study we used ZGP in two different strategies. First: ZGP, trained alone on a subset of pixels from each image of the training set. The descriptors of the pixels are their RGB values and the model is trained to classify each pixel of the image as cloud or no cloud. Second: FCN+ZGP, where ZGP is trained to classify the segmentation map resulting from a FCN In both cases, the resulting model corresponds to a mathematical expression which, applied to the new pixels of either the image or the segmentation map, returns the value of the predicted class (cloud or no cloud).

3. THE OPS-SAT IMAGES DATASET

3.1. Preliminary OPS-SAT data set

When performing supervised learning, the fist task consists in collecting and labelling data to train and test the AI algorithms . Without any actual OPS-SAT images available, we first use exogenous data from publicly available cloud segmentation datasets to train and test the models before the launch of OPS-SAT. These training data set included images from the Cloud-38 [9] and CloudPeru2 data sets[12].
Preliminary results on actual OPS-SAT images have been published in [1] showing quite good results considering the exogenous data used for training.

3.2. The OPS-SAT training

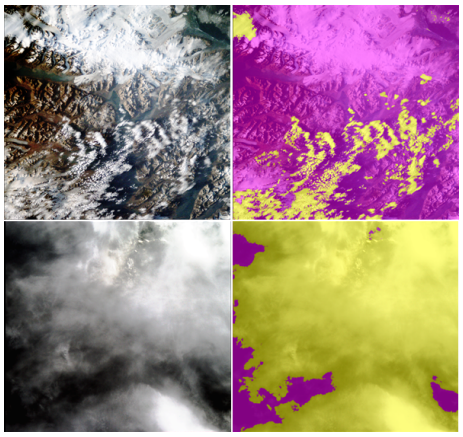

Figure 2: Examples of OP-SAT images. Top: sample from the training data set with cloud and snow. Bottom: sample from the test data set showing saturated and dark clouds. Left: RGB image, right: cloud mask overlaid on image (yellow=cloud, magenta=no cloud)

From the first OPS-SAT acquisitions provided by the OPS-SAT teams at the end of August 2020, a subset of 19 scenes have been selected in order to be representative of the diversity of OPS-SAT image acquisitions conditions (off-nadir angle, sun elevation, latitude) with different cloud coverage (fully cloudy, partially cloudy with sparse and large clouds as well as cloudy-free scenes) and including various landscapes such as forests, deserts, relief, water and snowy areas as shown in Figure 2. These 19 images have been annotated at pixel level by an image processing expert using a semi-manual labelling technique. This produced a binary cloud no-cloud segmentation mask usable as ground truth to fine tune and test the FCN on actual OPS-SAT images. For CNN, non-overlapping patches of size 28x28 have been extracted from each annotated image, leading to a set of 95 703 patches and class pairs. These patches have been classified according to their cloud coverage from 0% (cloud free patch) up to 100% (fully cloudy patch) by step of

10% cloud coverage in order to build a ground truth database usable to tune and validate the CNN and SNN. Then to avoid biases, the ANNs were trained on a reduced dataset, excluding for instance images difficult to annotate with sufficient confidence. At the end only 15 images were kept for training. Regarding the full ZGP model, the algorithm has been trained on a subset of pixels equally sampled from these 15 images leading to a full dataset containing 10000 pixels.

3.3. The OPS-SAT in-flight test data set

On March 2021, the CIAR experiment have been activated on-board OPS-SAT. The Figure 5 shows the first on-board inference results obtained with the new CNN after fine-tuning. Even if at first glance, the predicted cloud map fits well the cloud location when overlaid on the original image (see Figure 5 on the right), one need to get ground truths to assess in-flight performance metrics. For that purpose the OPS-SAT team provides us with 23 new images that have been annotated by the same image processing expert with the same technique used to annotate the training database. It shall be pointed out that the expert did not have any knowledge of the on-board inference results on these 23 images when he performed the annotations. For both training and test data sets, the following rules have been applied while annotating OPS-SAT images. Transparent clouds such as cirrus or haze are annotated as no cloud if some landscape features are visible trough. Very low radiometric level areas produced by clouds drop shadows are annotated as no cloud if no landscape features or cloud textures but noise are visible after image equalization. The test data set is composed of 23 pairs of images and binary cloud/no-cloud segmentation masks directly usable to assess the performance metrics of our AI algorithms. In flight test images encompass a large variability of acquisitions conditions, cloud coverage and landscapes including cloudy areas with saturated pixels and snowy landscape as shown Figure 2.

4. DEPLOYMENT ON OPS-SAT PAYLOAD CHIP

4.1. Deployment on HPS

The OpsSat's SoC FPGA Cyclone® V contains an ARM-based Hard Processor System, which runs Linux and a Java Virtual Machine (JVM). This JVM runs ESA NanoSat MO Framework (NMF): as stated in its website, NMF is a Java *software framework for small satellites based on CCSDS Mission Operations services. It introduces the concept of Apps in space that can be started and stopped from ground. Apps can retrieve data from the platform through a set of well-defined MO services.* We thus developed our experiment as an NMF *App* that subscribes to on-board services like Camera, GPS, Telecommand and exposes its own services like Telemetry. Our app can be run in realtime mode to process pictures provided service Camera then to send to Earth the results as a telemetry that embeds the compressed cloud segmentation map or only the cloud coverage percentage of the picture. However, in order to access pictures taken by other on-board experiments, our app can be run by ESOC team in offline mode to process pictures stored on-board in a specific SD-CARD directory. In both realtime and offline mode, our *App* reformats the RGB 4-mega-pixel picture, by tiling it into more than 5000 non-overlapping patches of size 28x28, writes it into the memory-mapped DDR-RAM of the FPGA, then reads the cloud segmentation map from the DDR-RAM.

4.2. Deployment on FPGA

The literature provides many tools to translate network information such as weights, biases and the network structure in HDL code which can be implemented on FPGA. We used an ANN to HDL generator that allows to use a "pipelined" approach to minimize the execution time and to take advantage of the efficient FPGA parallel calculation. However, this approach constrains the number and the size of the layers so that the entire neural network shall fit into the FPGA. Let's focus on how the ANN is implemented in the Cyclone V SX and then how it has been integrated with a full design. As shown in Table 1, our ANNs are composed of different layers: 2-Dimensional Convolution layers, Max-pooling layers, and FC layers.

4.2.1. Workflow

To implement the ANN on the MitySom FPGA, we have to perform the following steps:

- Train the ANN and export its weights, with a python script converting the input file .h5 into an HDL package with quantified weights and biases.
- With the 'formal' CNNs, generate the base HDL code for the "CNN" module with VGT.
- Modify the generated code to integrate layers that are not supported by the tool, and add fixed point arithmetic.
- For the SNN, directly use our SNN dedicated HDL code.
- For the HNN, use VGT to generate the HDL for convolution and pooling layers, and integrate the dedicated HDL code for parallel fully-connected layers and 'formal' to 'spiking' interface.
- Validate the RTL code on a sub-part of the cloud test dataset with ModelSim.
- Integrate the NN module into a complete design with Qsys and synthesize the design.
- Test the synthesized design on the Cyclone V using the complete cloud test datasets.

4.2.2. Design Overview

The design implemented in the logic part of Cyclone V is represented in Figure 3. The "DDR controller" allows connection to the off-chip memory and the "Direct Memory Access" allows moving the data from/to the DDR. Three others components have been developed: the "CNN" module with the ANNs layers previously described. This architecture is detailed in [1].

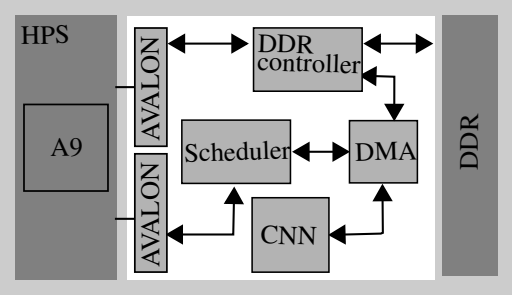

Figure 3: Design overview

4.2.3. Fixed-point arithmetic

After training, weights and biases are converted into fixed-point precision in order to save Logic occupation. To simplify the HDL design, we implement the same fixed-point arithmetic for each layer.Fixed-point arithmetic is composed of two parts: the integer part, which must be composed of enough bits to avoid overflow in the different layers, and the fractional part, which must be composed of as many bits as possible to have a precision as close as possible to the one of the training.
Thus a compromise must be found in order to avoid overflow but also to have enough precision. We tested several fixed-points configurations FxP$X.Y$ whose X and Y bits are used respectively after and before the radix point. Our tests shows the best arithmetic compromise for the CNN is FxP8.9 since it improves by several points the FxP8.8, with limited lost with respect to the results in Full Precision. However the FxP7.10 arithmetic leads to overflows, which causes a strong degradation of classification performances.
A deeper analysis of the weights and biases, before and after our naive quantization without retraining, shows in Table 2 that many values have been rounded to zero, but this heavily depends on the network training and its weights and biases initialization. Quantization-aware training should not only alleviate this problem but also prevent underflow or overflow to occur.

4.2.4. On-ground preliminary results

The results of the implementation in terms of logic resources, power consumption estimation, latency and accuracy are presented in Table 2. The logic resources and power consumption estimations are extracted from the hardware synthesis report of Quartus II® software, while the accuracies and latencies are measured by running the application on the Cyclone V MitySom board. In current designs the parameters are directly stored in the programmable logic of the FPGA.
During quantization, weights that are too small become zero. The more weights are set to 0, the more area is saved. For instance, we can see in Table 2 that after quantization more than half of the CNN parameters are reduced to 0, what impacts its cloud detection efficiency dividing by more than two its feature maps size.
As expected, SNN experiment brings interesting logic resources savings, while increasing latency due to the time-based computation paradigm. The accuracy is also slightly affected by a trade-off between latency and accuracy in the SNN hyper-parameters selection, in order to mitigate latency increase.

ANN algorithm	FCN 8.11	CNN 8.9	HNN 8.8	SNN 8.8
Zeroed weights and bias (#)	0/0614	809/1440	8/1428	15/2666
Logic Cells occupation (%)	92	69	63	31
LUTs in classif. stage (#)	670	5149	1657	332
Registers in classif. stage (#)	374	3153	1035	339
Block Memory in classif. stage (Kbit)	0	0	3.12	36.35
Average latency per 28x28 patch (μs)	24.99	24.84	279.1	312.0

Table 2: Hardware resources usage and latency results of the formal and spiking architectures for the ANN topologies of Table 1.

5. IN-FLIGHT RESULTS ANALYSIS

5.1. Hardware figure

5.1.1. Code uploading

A first feat in Europe is the uploading on-board of OPS-SAT, by ESOC team, of the full CNN experiment. First, several experiments are tested and validated on-ground on a MitySOM-5CSx. An experiment consists of two parts: Fisrt, a software that monitors data transfers, and performs pre and post-processing. Second, a bitstream that describes FPGA configuration for a specific NN architecture.
Once a network has been validated, ESOC team tests the experiment under similar conditions than those encountered in-flight thanks to an engineering model. Then, experiment is sent to the satellite with an average measured data rate of 12,818kB/s using Band S. The upload consists of two compressed files with IPK extension. First, the software to be executed is a 3.3MB IPK file. It contains Java code and jar librairies. This 3.3MB IPK file is

split into four binary chunks each of size 826kB. Each of these four chunks takes 1 minute and 8 seconds to uplink. Then the FPGA Bitstream along with preloader, bootloader is compressed and packaged into an IPK of 1.5MB in total size. It takes 1 minute and 46 seconds to uplink this file to the satellite. This first demonstrates the possibility of exploiting the reprogramming capabilities of FPGAs to remotly enhance Artificial Intelligence, or to change the mission of satellites after launch.

5.1.2. Execution time

In-flight inference in the FPGA part of an image of 2048x1944 pixels is done in less than 156ms for the CNN and FCN. In total, it takes 4 seconds to process a large image with the CNN architecture, including all the pre-processing and post-processing performed in the software. The Table 3 highlights the limited post processing required by CNN with most operations made on FPGA. In the case of the FCN, there are 16 times more outputs to post-process in software than with the CNN. The processing is therefore longer than for the CNN. The addition of a ZGP post-processing to the FCN does not significantly impact the global inference time, since this classification operation is done in parallel with the FCN post-processing. The performances of the full ZGP solution could have been divided by two by using 2 threads on the dual core ARM Cortex A9 instead of 1.

Algorithm	Hardware	FPGA (ms)	Total (ms)
CNN 8.9 No pre-processing	FPGA	155	4 370
FCN 8.11	FPGA	156	12 106
FCN +ZGP	FPGA +CPU	156	12 177
ZGP	CPU	0	32 411

Table 3: Time to process 5037 28x28 patchs including the post (e.g. to crop the image) and pre processing (e.g. to generate the telemetry) operations made on HPS.

5.1.3. Power consumption

ESOC provides us current telemetry of two Power Distribution Units(PDU). Each PDU have +5V lines going to Satellite Experimental Processing Platform(SEPP). One is dedicated to power the HPS part, the other to power the FPGA part. Each experiment consumption is detailed in Table 4.
Power consumption of the HPS part is 2.4 W in average for the CNNs. In the case of the FCN, the consumption of the HPS part is higher than for the CNN. This difference is explained by the fact that the ZGP-based algorithms are executed in parallel to the FCN pre and post-processings. The average power consumption of the FPGA part is around 1.7 W when inferring images with CNN architectures. In contrast, image inference with the FCN architecture represents an average power consumption of 1.61W. This is 140mW less than the estimation of Quartus Power Analyzer.
Finally, it can be seen that the total power consumption of the different experiments performed on-board is rather stable and represents more than 4W in average.
In the future, it will be interesting to analyze the on-board power consumption of an experiment based on SNN architecture in order to identify power consumption gains due to the architecture.

ANN architecture	HPS DPU	FPGA DPU		Total power
		Average	Quartus estimate	
CNN 8.8	2,38±0,12	1.70± 0.12	1.59	4.08± 0.19
CNN 8.9	2.41±0.11	1.68±0.09	1.59	4.09± 0.17
FCN 8.11 + Hybrid ZGP + ZGP	2.61±0.08	1.61±0.05	1.75	4.22±0.08
SNN 8.8 (*)	-	-	1.06	-
HNN 8.8 (*)	-	-	1.32	-

Table 4: In-flight power consumption in Watt. (*) means that these results have been obtained on-ground on the same hardware of the on-board OPS-SAT.

5.2. Inference metrics in-flight

As explained in Section 3, ESOC team gathered 23 4-megapixel pictures that we annotated on-ground while they were processed on-board by our algorithms. ESOC then sent us the segmentation map of each picture for each algorithm, which enabled us to compute the following metrics used in Table 5.

- Accuracy: 100 * (TP + TN) / (TP + TN + FP + FN)
- F-score : 2 * precision * recall / (precision + recall)

TP and TN stand respectively for true positives and negatives; FP and FN stand respectively for false positive and negative. Among the 23 images acquired on-board, we selected 10 images to constitute our test dataset evaluation while balancing classes cloud/no-cloud, and ensuring diversity in terms of texture (land, snow, desert, mountains, cirrus, cumulo-nimbus, etc) and radiometry. To compare the metrics , the same approach was applied on the training images, selecting 10 images in order to get a balanced dataset.

A first quantitative analysis of the Table 5 highlights:

- CNN, HNN and SNN metrics are degraded on test dataset (negative β-α results) that could be explained by overfitting. However the improvement of the FCN metrics can highlight a difference in the train and test images texture or radiometry impacting the results (to be further investigated)
- CNN with lowest overall metrics on training dataset (the first ANN deployed on OPS-SAT) is degraded but not so much by zeroed weights and biases on

ANN algorithm	Training-set on GPU (α)			Testing-set on GPU (β)						Testing-set on FPGA (σ)					
	Fscore	Preci	Recall	Fscore	β-α	Preci	β-α	Recall	β-α	Fscore	σ-β	Preci	σ-β	Recall	σ-β
CNN (8.9)	58	74	47	56	-2	70	-4	46	-1	49	-7	76	6	37	-9
SNN (8.8)	62	79	51	53	-9	69	-10	44	-7	67	14	62	-7	72	28
HNN (8.8)	61	80	49	58	-3	70	-10	50	1	67	9	76	6	59	9
FCN (8.11)	62	81	51	72	10	75	-6	70	19	72	0	74	-1	69	-1
FCN+ZGP	60	78	49	71	11	74	-4	68	19	71	0	74	0	68	0
ZGP (FP32)	60	62	58	63	3	65	3	62	4	63	0	65	0	62	0

Table 5: Quality metrics of the different AI algorithms when executed on GPU with a 32-bits Floating-Point precision on training (α) and testing (β) OPS-SAT datasets, whose comparison highlights the generalization capability, and the inferences of the testing (σ) OPS-SAT dataset on FPGA to highlight the impact of the Fixed-Point quantization when compared to inferences on GPU/FP32 (β). Results on FPGA (σ) are in-flight inferences except for HNN and SNN, not yet uploaded on OPS-SAT.

board quantization (σ-β results). However its Precision, that is a major metric for our use case (limited risk to discard images without cloud), is the best one, with worst recall.

- SNN is impacted by spikes encoding, but the results are very good (F-score close to 70%) considering the limited resources needed on FPGA (around 30%).
- HNN F-score on FPGA is also close to 70%, with best Precision but limited Recall performance.
- HNN and SNN have overall metrics (mainly the Recall) increased on FPGA (α-β). Both ANN with spiking layers implement a 'spike generator'. It transforms the static image' pixels to spike trains, permitting a better representation of data which reduce uncertainties at the output of the network. The SNN's architecture is presented with more details in a thesis manuscript that is available in [6].
- FCN is clearly the best ANN thanks to 16 more outputs. This network is not impacted by quantization (σ-β results) without any zeroed parameters. It also takes benefit of the full FPGA resources (more than 90%). Its main drawback is difficulty to segregrate snow with cloud class.
- ZGP provides homogenous results with very good generalization (β-α results), and no quantization issues since it is running on CPU.

A qualitative analysis of the full precision images with False Positive and False Negative reported shows that all algorithms detection are pertinent. However as expected it is difficult to segregate snow and cloud for most of them except the CNN what explains its good precision. In counterpart this ANN is poor on large clouds with radiometry variance.

6. FURTHER IMPROVEMENTS

During this experiment, we faced some challenges that hinders us to achieve better performances:

- Segmentation in RGB of clouds that are non-opaque objects, with undefined shapes, contrary to objects found in classical datasets (eg: animals, vehicules),
- Large radiometric variations from dark to bright clouds caused by a wide variability of Sun illumination conditions due to the OPS-SAT Polar orbit and large off-nadir pointing angles,
- Characterization of the camera, which is unavailable hence hinders us to calibrate the image radiometry,
- Small dataset, less than 20 images, including several scenes difficult to annotate,
- Capabilities of the MitySOM FPGA limiting the ANNs size and its parameters arithmetic.

Several actions are identified to improve the overall performances. To upgrade the ANNs metrics we propose several improvement. First, we intend to perform corrections of the raw data to compensate for the Sun illumination variations. Second, we propose to improve annotation method by adding an unknown class, typically black holes like drop shadows inside clouds, clouds borders or transparent clouds that can be ambiguous. Lastly, we foresee to increase the training dataset in quantity (tens of images) and in quality (more texture representativeness of the ground and the cloud diversity), with well-balanced classes.
Moreover, new architectures are under development in compliance with the MitySOM capability. An enlarged SNN architecture is under evaluation, implementing more filters or additional layers, to take benefit of the margin resources on the FPGA of the current SNN. Additionally, to take full control of ANN deployment on FPGA, we are developing a custom tool. It will parse .h5 model file and automatically generates HDL networks. These networks contain package for weights and top-level wrapper that properly instantiates and tunes the different parameters of the HDL layers. In this way, the CNN will be optimally implemented on FPGAs, managing also the weights and biases zeroed.
Finally, the ZGP solution can be trained on more pixels from the training images in order to get a more representative sample of the entire training set and therefore to

reduce the errors on some types of ground that have not been learnt by the current model. In order to consider the context of each single pixel and not only its RGB values, we plan also to introduce the values of the pixel's neighbors in a delimited window. Also, to reduce the computational processing time on HPS/CPU by two, a multithreaded implementation of ZGP on two threads can be performed.

7. CONCLUSION

This OPS-SAT experiment demonstrates the capability of embedded AI to infer images on-board satellite with a very high throughput (more than 25 millions of pixels per second) and a very low consumption (1.8 Watt in worst case on FPGA). Another success is the uploading on-board OPS-SAT, by ESOC team, of the full experiments: around 5min to upload HPS codes, including Java libraries, and around 2min to upload the FPGA bitstream.

So this successful demonstration changes the space observation paradigm with the opportunity to extract on board satellites only the useful information for the end users. This close real-time processing of the data by AI, with the capability to upload the algorithms in-flight, opens a lot of opportunities like performing alert from space, above all on small satellites, and re-programming missions in orbit.

ACKNOWLEDGMENTS

We first thank the European Space Agency for providing us the FPGA development kit and for supporting our experiments on OPS-SAT, in particular David Evans (the mission manager), Tom Mladenov (the operations manager) and Vasundhara Shiradhonkar (our experiments support). This work is part of the CIAR Project from IRT Saint Exupery, thus we thank also all our partners: ActiveEon, ADVANS GROUP, GEO4i, Inria, LEAT/CNRS, MyDataModels, Thales Alenia Space, TwinswHeel, and Thales Research & Technology collaborating on the spiking neural networks development.

REFERENCES

1. F. Feresin, M. Benguigui, Y. Bobichon, E. Lemaire, M. Moretti, and G. Bahl, "On board images processing using ia to reduce data transmission: example of opssat cloud detection," in 7th On-Board Payload Data Compression Workshop, OBPDC, 2020.

2. S. Ji, P. Dai, M. Lu, and Y. Zhang, "Simultaneous cloud detection and removal from bitemporal remote sensing images using cascade convolutional neural networks," IEEE Transactions on Geoscience and Remote Sensing, pp. 1–17, 2020.

3. N. Abderrahmane, E. Lemaire, and B. Miramond, "Design space exploration of hardware spiking neurons for embedded artificial intelligence," Neural Networks, 2019.

4. L. Khacef, N. Abderrahmane, and B. Miramond, "Confronting machine-learning with neuroscience for neuromorphic architectures design," in 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–8, IEEE, 2018.

5. A. N. Burkitt, "A review of the integrate-and-fire neuron model: I. homogeneous synaptic input," Biological cybernetics, vol. 95, no. 1, pp. 1–19, 2006.

6. N. Abderrahmane, Hardware design of spiking neural networks for energy efficient brain-inspired computing. Phd thesis, Université Côte d'Azur, Dec. 2020.

7. O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," CoRR, vol. abs/1505.04597, 2015.

8. G. Bahl, L. Daniel, M. Moretti, and F. Lafarge, "Low-power neural networks for semantic segmentation of satellite images," in Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops, Oct 2019.

9. S. Mohajerani and P. Saeedi, "Cloud-net: An end-to-end cloud detection algorithm for landsat 8 imagery," CoRR, vol. abs/1901.10077, 2019.

10. S. Mohajerani, T. A. Krammer, and P. Saeedi, "A cloud detection algorithm for remote sensing images using fully convolutional neural networks," in IEEE 20th International Workshop on Multimedia Signal Processing (MMSP), pp. 1–5, 2018.

11. A. Boisbunon, C. Fanara, I. Grenet, J. Daeden, A. Vighi, and M. Schoenauer, "Zoetrope genetic programming for regression," in 2021 Genetic and Evolutionary Computation Conference (GECCO'21), 2021.

12. G. Morales, S. G. Huamán, and J. Telles, "Cloud detection in high-resolution multispectral satellite imagery using deep learning," in Artificial Neural Networks and Machine Learning (V. Kůrková, Y. Manolopoulos, B. Hammer, L. Iliadis, and I. Maglogiannis, eds.), (Cham), pp. 280–288, Springer International Publishing, 2018.