CloudSatNet-1: FPGA-based Hardware-Accelerated Quantized CNN for Satellite On-Board Cloud Coverage Classification

Radoslav Pitonak[1], Jan Mucha[2], Lukas Dobis[1], Martin Javorka[1], Marek Marusin[1]
[1]Zaitra, Technicka 2935/23 616 00, Brno, Czech Republic
[2]Department of Telecommunications, Brno University of Technology
Technicka 12, 61600 Brno, Czech Republic

CubeSats, the nanosatellites with a wet mass up to 10 kg, accompanied by the cost decrease of accessing the space, amplified the rapid development of the Earth Observation industry. Acquired image data serve as an essential source of information in various disciplines like environmental protection, geosciences, or the military. As the quantity of remote sensing data grows, the bandwidth resources for the data transmission (downlink) are exhausted. Therefore, new techniques that reduce the downlink utilization of the satellites must be investigated and developed. For that reason, we are presenting CloudSatNet-1: an FPGA-based hardware-accelerated quantized convolutional neural network (CNN) for satellite on-board cloud coverage classification. We aim to explore the effects of the quantization process on the proposed CNN architecture. Additionally, the performance of cloud coverage classification by biomes diversity is investigated, and the hardware architecture design space is explored to identify the optimal FPGA resource utilization. Results of this study showed that the weights and activations quantization adds a minor effect on the model performance. Nevertheless, the memory footprint reduction allows the model deployment on low-cost FPGA Xilinx Zynq-7020. Using the RGB bands only, up to 90 % of accuracy was achieved, and when omitting the tiles with snow and ice, the performance increased up to 94.4 % of accuracy with a low false-positive rate of 2.23 % for the 4-bit width model. With the maximum parallelization settings, the hardware accelerator achieved 15 FPS with 2.5 W of average power consumption (0.2 W increase over the idle state).

Index Terms—**CNN; FPGA; hardware accelerators; image processing; on-board processing; quantization;**

I. Introduction

OVER the last decade, the Earth Observation (EO) industry has experienced a dramatic decrease in the cost of accessing space [1]. With the introduction of nanosatellites (a small satellite with a wet mass between 1 to 10 kg), known as CubeSats [2], the rapid development of remote sensing technologies was amplified [3]. As of 2021, more than 1500 CubeSats have been launched [4], and according to [5], it will increase up to a thousand satellites per year till 2028. Naturally, as the number of satellites grows, satellite imagery becomes readily available. Harvested data plays a significant role in various disciplines like environmental protection, agriculture engineering, land or mineral resource exploration, geosciences, or military reconnaissance [6], [7]. In line with the amount of remote sensing data acquired, the bandwidth resources for the data transmission inclines to be overloaded. Therefore, new techniques for efficient bandwidth resources management must be investigated and developed.

Several studies estimate that approximately $67\,\%$ of the Earth's surface is covered with clouds [6], [8], [9]. Consequently, most of the remote sensing imageries (RSI) will be contaminated by them, which devalues the quality of RSI and negatively affects the post-processing [6]. Cloudy conditions impair satellite sensor capabilities to obtain the clear views of the Earth's surface, and hence the quick and accurate detection of the cloudy images is necessary [6], [10], [11]. In general, the current methods for cloud coverage estimation or classification are mainly categorized to traditional and intelligent approaches [12]. Traditional ones consist of threshold-based (fixed or adaptive), time differentiation, and statistical methods. The threshold-based approaches rely on a visible reflection and infrared temperature of the clouds, therefore its performance weakens on low-contrasted (cloud vs. surface) images [13]–[15]. Time differentiation methods effectively identify the changing pixel values as clouds in multi-temporal images, however, they do not consider changes in the top of atmosphere reflectance affected by floods [12], [16]. Statistical methods combine spectral and spatial features extracted from RSIs with classical machine learning algorithms (support vector machine, decision tree), but they lack to obtain the desired results [17], [18]. To sum up, traditional methods provide some capabilities of cloud detection, though, they are susceptible to the backgrounds, are non-universal and subjective [12].

A more efficient approach to cloudy images detection comprises convolutional neural networks (CNNs), simple linear iterative clustering, or semantic segmentation algorithms [12]. Especially attractive are CNNs, which provide state-of-the-art results for many different tasks, including image classification, segmentation, and object detection. This success is often achieved thanks to models with a huge number of parameters which means large size and limited ability for the deployment on resource-constrained hardware. In recent years, there has been a tendency to deploy these models in line with the edge computing paradigm on resource-constrained hardware [12], [19]–[21]. Various hardware accelerators are available on the market ranging from microcontrollers for smaller models to boards equipped with GPU, visual processing unit (VPU), or field-programmable gate array (FPGA). FPGA in particular provides interesting capabilities in terms of cost, flexibility, performance, and power consumption. A possible disadvantage is a longer time to market in comparison to GPU or VPU solutions. Nevertheless, this gap is being closed by recent advancements in the hardware deployment of machine

learning models [22], [23] created in well-known machine learning frameworks like Pytorch or Tensorflow. Considering the payload limitations of the CubeSats, the optimal solution of the CubeSat's cloud detection system is a system estimating RSI cloud coverage running directly on board. To reduce the costs and development time of such real-time detection systems, Commercial-Off-The-Shelf (COTS) components provide a favorable deployment option [24]. The crucial criterion for an onboard detection system is its power consumption, whereas the usual limit is below 5 W and the ratio of the falsely discarded images below 2 % [12], [19], [20].

In line with the above mentioned, Zhang et al. [25] introduced a lightweight CNN for cloud detection based on U-Net using red, green, blue, and infrared waveband images from the Landsat-8 dataset. Applying the LeGall-5/3 wavelet transform (4 levels) for dataset compression and processing time acceleration, the authors reported 94.3 % of overall accuracy running on an ARM-based platform. Similarly, in [26], the authors applied depthwise separable convolutions to compress the model of U-Net and accelerate the inference speed. The Study reported the best accuracy of 90.54 % verified on Landsat 8 remote sensing images. Another utilization of a lightweight MobU-Net trained on Landsat 8 dataset and using JPEG compression strategy was performed by [27]. The achieved overall accuracy was around 93.1 % for a model deployed on ARM9 processor on Zynq-7020 board. Maskey et al. [3] proposed an ultralight CNN designed for on-orbit binary image classification called CubeSatNet. The model was trained on BIRDS3 satellite images and deployed on ARM Cortex M7 MCU. An accuracy of 90 % was achieved when classifying images as "bad" for cloudy, sunburnt, facing space, or saturated images and "good" in all other cases. A promising method for cloud detection using RS-Net and RGB bands exclusively was published by [28]. For model training, the Sentinel-2 dataset was used, and 76 % of accuracy was reported by the model deployed on an ARM-based platform. Another possibility is to use the Forwards Looking Imager instrument, which provides analysis of the upcoming environment of the satellite. Tnot found, testing various lightweight CNNs deployed on the Zynq-7020 board using FPGA. The authors reported high accuracy of 98 %, however, 100 images only were used for testing. To sum up, lightweight CNNs provide a competitive on-board cloud detection performance in comparison to the state-of-the-art deep convolutional neural networks, like CDNetV1 [6] or CDNetV2 [10]. CDNetV1 is a neural network for cloud mask extraction from ZY-3 satellite thumbnails with the accuracy of 96.47 % [6]. Its extended version, CDNetV2, focuses on adaptively fusing multi-scale feature maps and remedying high-level semantic information diluted at decoder layers to improve cloud detection accuracy with cloud-snow coexistence. The authors confirmed the robustness of the proposed method using validation on several other datasets like Landsat-8 or GF-1.

To the best of our knowledge, the CloudScout cloud detection method proposed by Giuffrida et al. [29] and later extended by Rapuano et al. [20] is the most related work to this study. The method was developed in the frame of the Phisat-1 ESA mission, which exploits a hyperspectral camera to distinguish between the clear and cloud-covered images. To reduce the bandwidth, the mission has set a criterion that only images that present less than 70 % of the cloudiness are transmitted to the ground. CloudScout was trained using Sentinel-2 hyperspectral data and achieved the 92 % of accuracy, 1 % of false positives with the power consumption of 1.8 W deployed on re-configurable Myriad-2 VPU by Movidius Intel [29]. Nevertheless, the authors identified multiple drawbacks due to the Myriad-2 design, which is not specifically suitable for the space environment (not based on a radiation-tolerant technology) [20]. Therefore, the authors extended their work and proposed an FPGA-based hardware accelerator for CloudScout CNN. The authors compared the Myriad-2 VPU with two FPGA boards: Zynq Ultrascale+ ZCU106 development board and Xilinx Kintex Ultrascale XQRKU060 radiation-hardened board. Results obtained by Zynq Ultrascale+ ZCU106 show that the FPGA-based solution reduced the inference time by 2.4 times (141.68 ms) but at the cost of 1.8 times greater power consumption (3.4 W) [20]. Inference time estimated for the Xilinx Kintex Ultrascale XQRKU060 board was 1.3 times faster (264.7 ms) in comparison with the Myriad-2 device, however, the power consumption was not reported.

Regarding the presented achievements of the related works and trends in the CubeSats development, we may expect a new era of smart nanosatellites equipped with reconfigurable, programmable hardware accelerators with an on-demand edge computing paradigm at payload level [3], [12], [19], [20], [25]–[27], [29], [30]. A usual aspect of the presented studies is the employment of multispectral or hyperspectral RSI for the cloud detection system. Generally, the bands' composition of multi/hyperspectral RSI differs for individual missions, yet all are equipped with an RGB camera. Therefore, a cloud detection system built on RGB bands only may provide better portability for various missions independent of its multi/hyperspectral bands. In addition, the RGB cameras are several times cheaper and more convenient for short-term CubeSats missions. To the best of our knowledge, we identified only three studies [20], [21], [30] that performed deployment and evaluation of the CNN-based cloud detection method on an FPGA-based platform. Hence, in the scope of this study, we would like to present **CloudSatNet-1**: an FPGA-based hardware-accelerated quantized CNN for satellite on-board cloud coverage classification. More specifically, we aim to:

- explore effects of quantization introduced to the proposed CNN architecture for cloud coverage classification,
- investigate and optimize the performance of cloud coverage classification by biomes diversity and its false-positive identifications,
- explore hardware architecture design space to identify optimal FPGA resource utilization.

The rest of the paper is organized as follows. Section II describes the used dataset and its preprocessing. Methodology is described in Section III. In Section IV the results are summarized. The discussion can be found in Section V and the conclusions are drawn in Section VI.

II. Dataset

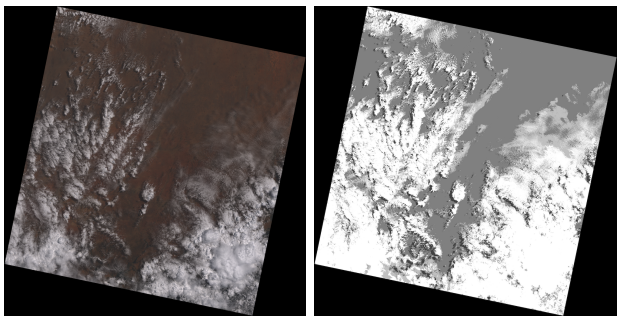
A. Data

For the purpose of this study, the Landsat 8 Cloud Cover Assessment Validation data (L8-D) [31] was used. The L8-D offer a balanced cloud distribution and diverse sets of land and water cover, which makes it a suitable source of data for the proposed CNN-based classification model. The L8-D was acquired by the Landsat 8 Operational Land Imager (OLI) and Thermal Infrared Sensor (TIRS) [32]. Furthermore, data are orthorectified and corrected for terrain relief using Level-1T processing [33].

The L8-D consists of 96 scenes divided into 8 biomes. The scene size is 185 km by 180 km, and each scene contains 11 multispectral bands with a resolution of 30 meters per pixel (except bands 8, 10, and 11, which are not used in this work). Manually annotated cloud coverage is stored as a cloud validation mask. The cloud validation mask is an image whose pixel values contain information about the level (or class) of cloudiness, interpreted using the following Table I. The example of the scene image (natural color composition) from the L8-D dataset can be found in Figure 1a, with its respective cloud mask in Figure 1b.

TABLE I: Interpretation of L8-D cloud mask pixel values [31].

Value	Interpretation
0	Fill
64	Cloud Shadow
128	Clear
192	Thin Cloud
255	Cloud



(a) Image patch (b) Cloud validation mask

Fig. 1: L8-D image patch example (left) reconstructed from bands B4, B3, and B2 with its associated multi-class cloud mask (right) [31].

Two cloud mask classes (thin cloud, cloud) are categorized as cloud pixels. From these pixels, the Cloud Cover Assessment (CCA) is computed as a ratio of cloud pixels to all pixels with values expressed in percentage [31]. The average CCA value for one scene is $48.35\,\%$. The distribution of the CCA values of L8-D scenes is shown in Figure 2. Scenes are categorized by their area of capture into biome classes by the International Geosphere-Biosphere Programme [34] into 8 following biomes: Barren (BA), Forest (FO), Grass/Crops (GC), Shrubland (SH), Snow/Ice (SI), Urban (UR), Water (WA), Wetlands (WE). They are distinguishable from each other by

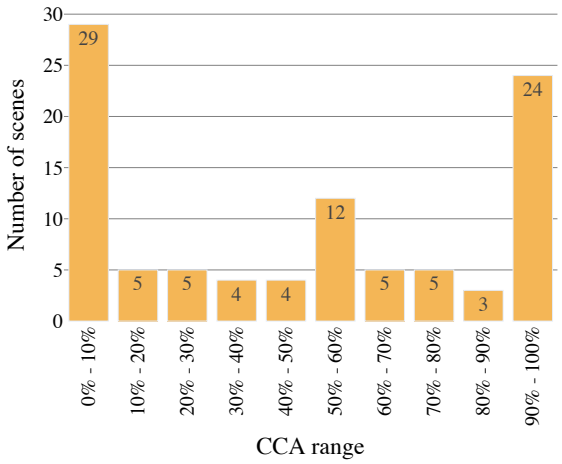


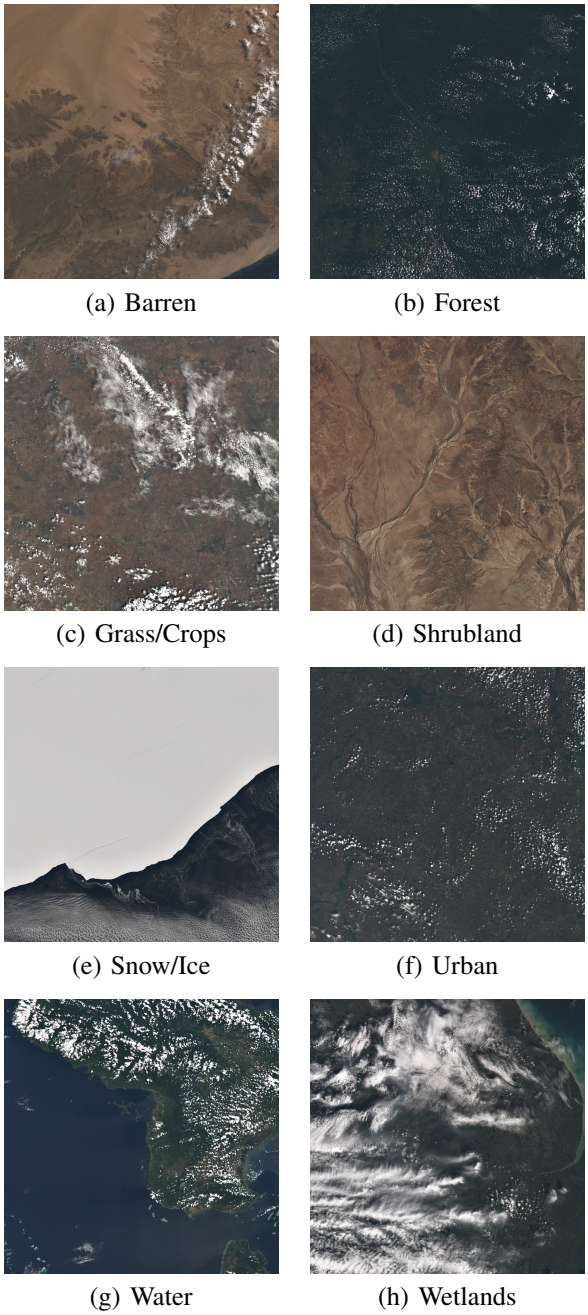
Fig. 2: Distribution of the L8-D scenes CCA values [31].

their visual properties, and they have various intensities of cloud to terrain contrast, which leads to different challenges for the cloud detection system working with RGB data. For example, the biomes with sharp cloud to terrain contrast, like Grass/Crops, have a large value of the derivative at the transition between terrain and cloud. Other biomes like Snow/Ice have a terrain with cloud-like features, which may lead to a large number of false positives in classifier predictions as their terrain blends with clouds. Examples of image patches for each biome of L8-D are shown in Figure 3.

B. Data Preprocessing

The image patch for each scene is a natural color composite from the combination of bands B4 (red), B3 (green), and B2 (blue). Values in patch images are re-scaled from the range 0-65535 to 0-255 using a MinMax normalization. Patch images in L8-D are georeferenced. The orbit path of the Landsat-8 does not go straight from south to north. The scene acquisition follows the orbit path of the satellite. Therefore the image appears to be rotated or tilted, like in Figure 4a. Redundant georefencing information can be neglected, when detecting clouds from satellite images. Next, the black (no-data) parts of the image need to be removed. The removing of the black parts consists of two steps. First, the image is rotated, so the actual image data are parallel to the whole scene image, as shown in Figure 4b. The rotation is using a nearest-neighbor interpolation method. Then, the image is cropped to lower resolution (from approx. 8000x8000 to approx. 6400x6400), so only image data are preserved, as illustrated in Figure 4c.

Image patch (with dimensions approx. 6400x6400x3) is cropped to 512x512x3 tiles, according the white lines in Figure 4c, omitting tiles at the edge that do not have full resolution. Each patch has a slightly different resolution after cropping, which causes a different number of generated tiles per patch (approx. 140). From 8 biomes each containing 12 scenes there are in total 13525 tiles. The original CCA values for the scene from Figure 2 do not apply to individual tiles. Generated tiles usually cover cloudy or cloud free areas. This



(a) Barren (b) Forest

(c) Grass/Crops (d) Shrubland

(e) Snow/Ice (f) Urban

(g) Water (h) Wetlands

Fig. 3: Image patches examples for each biome in L8-D [31].

generates a less tiles with balanced cloud coverage (or CCA value) in the final dataset (trade off for creating many tiles from fewer image patches). Tiles with CCA $\geq 70\,\%$ are categorized as cloud and the rest is categorized as non cloud tiles. Each of the 13525 tiles has assigned a corresponding binary cloud coverage label. To preserve the evenly distributed cloud coverage in the train, validation and test dataset, the tiles from a single image patch are divided into 5 CCA buckets: $0-20\%$, $20-40\%$, $40-60\%$, $60-80\%$, $80-100\%$. The distribution of the tiles and its CCA values per biome for the full L8-D dataset is visualised in Figure 5. The tiles from each patch CCA bucket are divided to train, validation and test dataset in the ratio 2:1:7, with the coherent variation of the biomes and their CCA values, as visualised in Figure 6. In this study, the reliability of the results and the model portability are prominent. Therefore, the testing dataset is dominant in comparison to the training or validation dataset. Moreover, more than 2700 tiles are considered a satisfactory quantity for the model training. Since the variation of the train, validation, and the test dataset is coherent, suppression or advantage of any of the biomes or the CCA bucket during the model training is not expected.

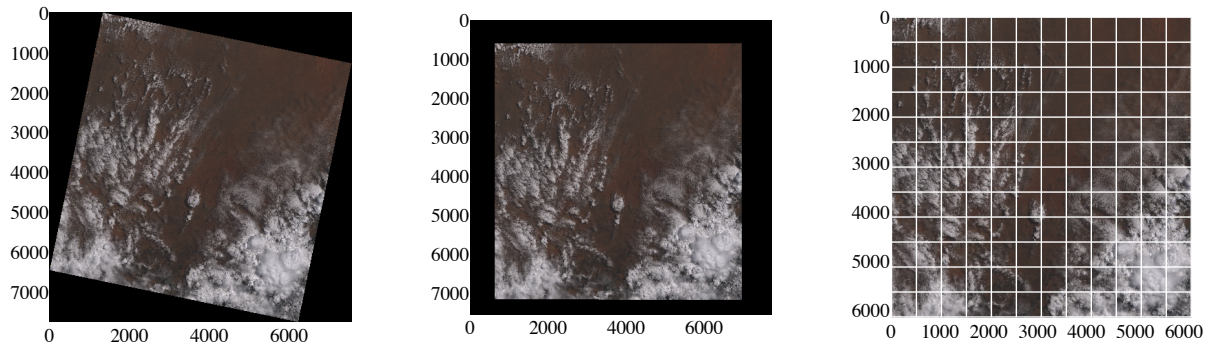
III. Methodology

The procedure is divided into three stages. First the baseline model of CNN with floating point parameters is trained. Then the weights and activations of the model are quantized and the model is re-trained. The last step is the deployment of the model on FPGA to achieve high throughput and low power consumption suitable for on-board data processing on satellite. To be able to deploy a CNN on the edge there are many techniques how to reduce the model memory footprint such as prunning or quantization. In this work the focus of interest is on quantization which replaces floating point operations and weight tensors with lower bit widths what is especially useful for FPGA where arbitrary precision data types can be implemented.

A. Quantized CNN

Quantization in neural networks is a technique used for optimization which proved to produce great success in the recent years [35]. Its main focus is on reducing memory footprint and computation time by replacing floating point compute operations and storing of tensor weights with lower bit widths. This is especially useful for resource-constrained applications. There are two ways how to introduce quantization to a neural network. The first one is to train the neural network with quantized parameters and the second one is a quantization of parameters after the model is trained with floating point precision. In the former case the process is called Quantization-aware training (QAT), in the latter it is referred to as Post-Training Quantization (PTQ). PTQ may disturb the model parameters and change the point to which it converged during the training with floating point precision. For this reason QAT is used for the experiments conducted in this study and training with quantized model parameters is performed. For more comprehensive review about the current state of quantization in neural networks refer to the recent survey [35].

The network was implemented using the Brevitas framework. Brevitas is a PyTorch library used for QAT of neural networks [36]. At the time of writing the PyTorch library supports the quantization as well but allowing just reduction from 32-bit floating point to 8-bit integer [37]. Brevitas in comparison allows to reduce weight and activation bit widths to as low as 1-bit which enables to create Binary neural networks (BNN) [38]. Another reason why the Brevitas library is used is that a model trained using Brevitas can be exported and used by the FINN framework for dataflow architecture acceleration (DFA) on Xilinx FPGAs [23]. FINN framework is a compiler



(a) Image patch (b) Rotated patch (c) Cropped patch with tiling

Fig. 4: L8-D scene during different preprocessing steps [31].

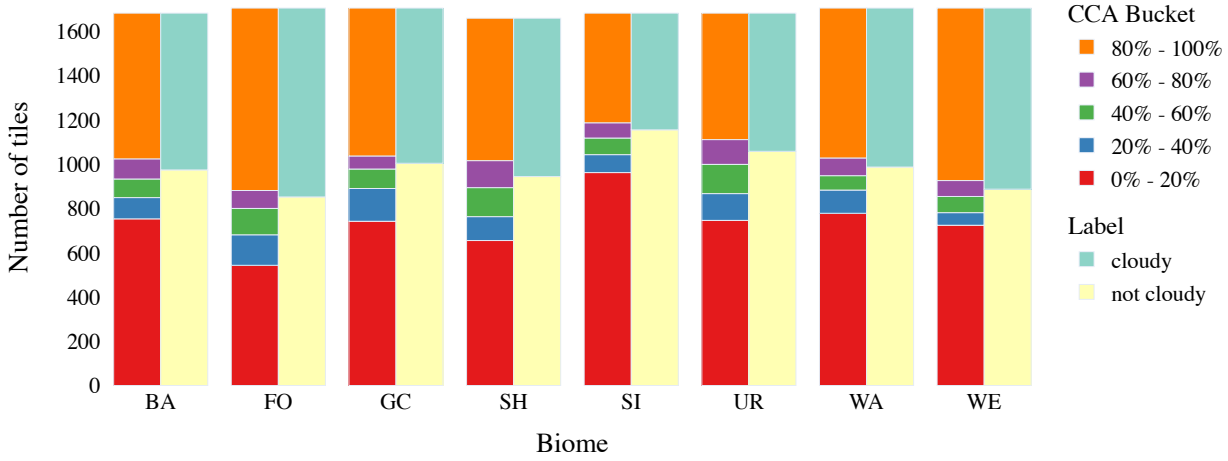


Fig. 5: Distribution of the tiles and its CCA values per biome for the full L8-D dataset. Tiles with CCA $\geq 70\,\%$ are categorized as cloudy and the rest are categorized as not cloudy tiles.

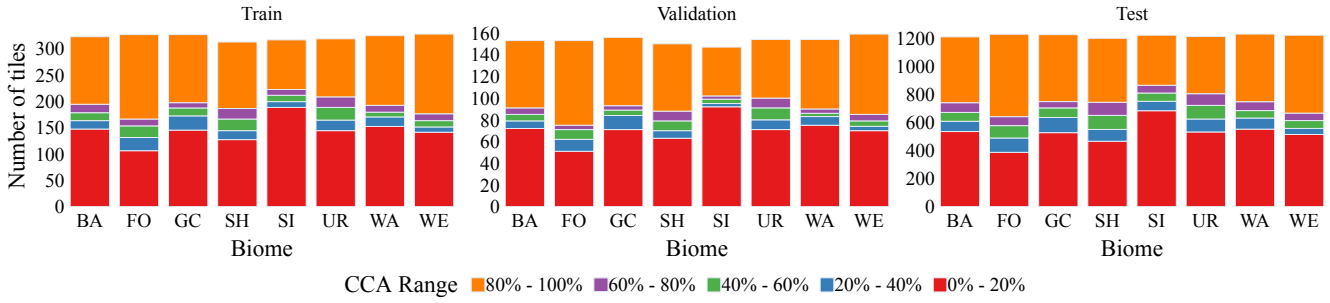


Fig. 6: Distribution of the tiles and its CCA values per biome for the training, validation and testing dataset.

for feed forward DFA for deep neural networks (DNN) inference. When DFA is used, every layer of DNN is mapped to its own set of dedicated compute and memory resources [39] which mimics the topology of DNN. In FINN the performance and resource usage can be controlled with a concept called *Folding*. FINN uses what is called Matrix-Vector Threshold Units (MVTU) for convolutional and fully connected layers. There are three parameters that can be set: Matrix-Vector Matrix-Multiple Vector (MMV) length, Processing Elements (PE), and SIMD (Single instruction multiple data) lanes. Using these parameters it is possible to control the throughput of the network with respect to resource utilization of the FPGA.

B. *CloudSatNet-1 architecture*

The proposed network architecture consists of 10 convolutional layers and 2 fully connected layers, their specific parameters are visualized in Figure 7. Each layer except the last layer uses the ReLU activation function and has

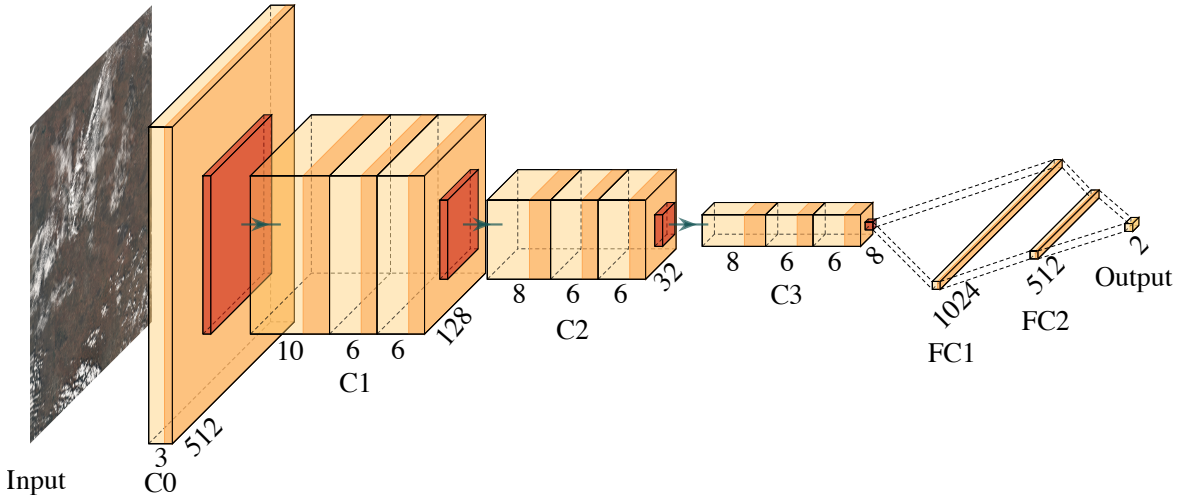


Fig. 7: CloudSatNet-1 architecture

no bias. The network starts with initial convolutional layer which processes 512x512x3 uint8 input and continues with 3 sequences of 3 layers each. The input size was chosen to allow direct comparison with CloudScout architecture [29]. Each sequence middle layer has lower number of filters to implement bottleneck for better generalization properties. After each sequence and initial layer there is batch normalization and max pooling with kernel size 4, this leads to effective reduction of feature dimensions. Last fully connected layer outputs unnormalized probability for each class where the first class represents cloud presence below 70% CCA in the image and the second class signals presence of clouds above this threshold.

1) Loss function

The loss function used for training of the model was a modified binary cross entropy loss with increased penalty for false positives (FP) errors shown in Equation 1. Penalty for FP errors is multiplied by a parameter α which is inspired by the approach reported in [29] where the authors showed decrease in number of FP errors while keeping accuracy on acceptable value when parameter α was set to 2.

$$F(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\frac{1}{N}\sum_{i=1}^{\mathrm{N}} y_i \cdot \log(\hat{y}_i) + \alpha \cdot (1-y_i) \cdot log(1-\hat{y}_i), \quad (1)$$

where y is the ground-truth label, $\hat{y}$ is the predicted output of the network and α is a hyper-parameter to increase penalty for FP errors.

2) Quantization process

First, the model with floating point precision is trained as a baseline. After sufficient accuracy has been achieved weight and activation bit widths are progressively reduced and the change in accuracy is observed. To fit the model on FPGA and achieve high throughput with acceptable accuracy and low power consumption, in this paper the focus of interest are bit widths of hidden layers lower or equal than 4. In all experiments same bit widths are used for weights and activations. The first and last layer of neural network can be more sensitive to quantization [40]–[42], so they were quantized to 8 bits. Last fully connected layer has also a quantized bias term. It was observed by the preliminary experiments that it is important to adjust weight initialization according to selected weight bit widths.

The proposed architecture contains blocks of convolutional layers followed by batch normalization and ReLU. This sequence has the advantage of hardware implementation which FINN framework [23] utilizes and usage of batch normalization layer leads to faster convergence [43]. After the training it becomes fixed linear transformation during the inference. Brevitas does not provide a quantized alternative to PyTorch batch normalization layer, but FINN framework supports native PyTorch batch normalization. Since the threshold-based quantized activation (ReLU) is used, batch normalization is implemented using successive thresholding in the FINN framework thanks to the process called Streamlining [44]. This process shows how integer only operations can be used for forward pass of a quantized neural network layer with uniformly quantized weights and activations.

C. Selected hardware

The trained neural network model is deployed on Zturn development board equipped with SoC Xilinx Zynq Z7020. Thanks to FPGA, Zynq is able to provide a platform for computationally intensive processing, but at the same time meets power consumption requirements of the developed CNN. The target frequency is set to 100 MHz. For the power consumption measurements, the J7-t USB safety meter was used.

Xilinx Zynq is an all-programmable System-on-Chip (SoC), which consists of the dual-core ARM Cortex-A9 processor coupled with FPGA based on Xilinx 7-series FPGA architecture into a single integrated circuit [45]. ARM Cortex-A9 is connected by industry standard AXI interfaces, providing low latency and high bandwidth between processor and programmable logic. FPGA programmable logic consists of 85000 logic cells, 53200 Look-Up Tables (LUTs), 106400 Flip-Flops (FFs) and 4.9 Mb of block RAM (SoC data-sheet

at [46]). In addition, it also contains 220 Digital signal processing (DSP48E1) slices for high-speed arithmetic embedded into fabric logic in proximity with Block RAM components. The processor is capable of running Linux operating system with PYNQ [47] library which enables usage of Python programming language for programming both the processor and hardware libraries called overlays. Power consumption in idle state with booted Linux Ubuntu 18.04 was measured to be 2.32 W.

D. Proposed Workflow

The pipeline used to create the hardware accelerated CNN consists of the following steps. First the baseline floating point model is trained and evaluated to observe standard metrics such as accuracy, recall, precision and F1 score. Next QAT is used to train the quantized model, which is evaluated in the same way as the baseline model. In addition to this a smaller verification dataset with same distribution of tiles in the respective cloud cover ranges is created and consists of 380 tiles. Per tile evaluation is performed on this dataset and resulting logits from the last layer are saved for the model verification deployed on FPGA. Quantized model is exported to ONNX format [48] and transformed to High level synthesis (HLS) code using FINN framework. Model is then synthetized using Vivado Design Tools from Xilinx and the resulting bitfile is deployed to FPGA. Evaluation of deployed model with focus on observing hardware accelerator attributes is performed. Per tile evaluation on verification dataset is performed and resulting logits are statistically compared using T-Test to measure model distortion caused by deployment of the model on the edge. Workflow is summarized on the scheme displayed in Figure 8.

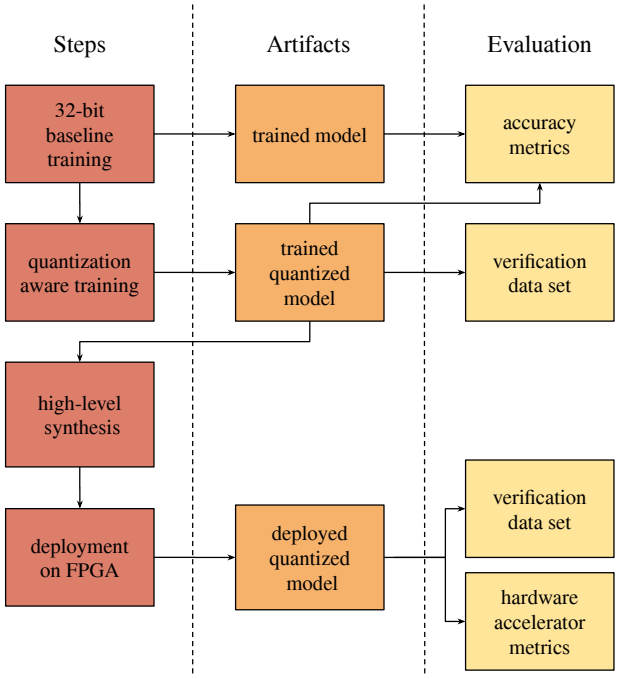


Fig. 8: Scheme of proposed workflow.

E. Experimental setup

The main goal is to experiment with end to end development of FPGA-based hardware-accelerated quantized CNN for on-board cloud cover classification. Therefore the experiments are divided into three stages: (1) Training of the classification model with focus on observing the impact of quantization to model accuracy; (2) Observe the accuracy of the resulting model on different biomes and remove the outliers from the dataset; (3) Explore hardware architecture design space to identify configuration with highest throughput, pre-defined target throughput and minimal FPGA resource utilization.

For the model training the aim is to achieve highest accuracy and minimize false positive rate (FPR) on test dataset for different bit widths of model weights and activations. Bayesian optimization (summarized in [49]) is used for hyper-parameter search of parameters defined in Table II. Number of epochs is set to 40 with early stopping when accuracy on the validation dataset starts to diverge. Overall 32 runs were conducted for each of 4 total configurations with hidden layers weight and activation bit widths set to 32, 4, 3, and 2. In both training scenarios, model performance will be evaluated on the test dataset using accuracy, precision, recall and F1 metrics with addition to FPR. In the second stage, the accuracy achieved on the particular biomes will be analyzed to identify the potential lack of the model performance. Regarding the achieved results, a new set of experiments will be conducted.

TABLE II: Hyper-parameter search constraints for Bayesian optimization during training of neural network.

Parameter name	Values
Learning rate	$\mathcal{U}(0.0005, 0.002)$
Learning decay	$0, 0.001, 0.0002$
Batch size	$32, 64, 128$
FP penalizer (α)	$\mathcal{U}(1, 4)$

In the last stage the hardware architecture design space is explored using FINN framework. Selection of parallelism in FINN can be defined as $P = MMV \times PE \times SIMD$ [50]. At the time of writing, FINN only supports MMV set to 1 so just PE and SIMD are used to increase parallelization in the experiments. The layer with the largest number of cycles will limit the overall throughput of the network [50]. The estimated number of clock cycles per layer for the proposed architecture is shown in Table III in two configurations. One with default folding (no parallelization) with the lowest performance and the second one with maximum folding achievable for the proposed architecture. First layer is the biggest bottleneck in the network so the DSP slices were assigned to it as it requires more resources to compute results with 8-bit inputs (uint8) and 8-bit weights.

IV. Results

Results of cloud coverage classification employing full L8-D dataset to train and evaluate proposed CloudSatNet-1 CNN are shown in Table IV. In the upper part of the table, most accurate models for each analyzed bit width (weight and activation) selected by ACC are presented. Top models for 32, 4, and 3-bit width provide similar classification performance

TABLE III: Parallelization settings and their respective estimated number of clock cycles

layer	Base folding (PE, SIMD)	Base folding Cycles	Maximum folding (PE, SIMD)	Maximum folding Cycles
Conv_1	(1,1)	70778880	(10,3)	2359296
Conv_2	(1,1)	8847360	(6,10)	147456
Conv_3	(1,1)	5308416	(6,6)	147456
Conv_4	(1,1)	7077888	(2,3)	1179648
Conv_5	(1,1)	442368	(1,1)	442368
Conv_6	(1,1)	331776	(1,1)	331776
Conv_7	(1,1)	442368	(1,1)	442368
Conv_8	(1,1)	27648	(1,1)	27648
Conv_9	(1,1)	20736	(1,1)	20736
Conv_10	(1,1)	884736	(1,2)	442368
FC_1	(1,2)	262144	(1,2)	262144
FC_2	(1,1)	1024	(1,1)	1024

[1] PE – processing elements; SIMD – single instruction multiple data; Cycles – estimated number of cycles; Conv – convolution layer; FC – fully connected layer

TABLE IV: Results of cloud coverage classification for best-performed models employing full L8-D dataset.

Top models by ACC					
BW	ACC [%]	PRE [%]	REC [%]	F1 [%]	FPR [%]
32	89.92	86.56	89.70	88.10	9.93
4	87.42	88.79	79.84	84.08	7.18
3	88.24	86.01	85.67	85.84	9.93
2	83.41	77.46	84.81	80.97	17.59
Top models by FPR					
BW	ACC [%]	PRE [%]	REC [%]	F1 [%]	FPR [%]
32	86.67	95.74	71.12	81.62	2.25
4	87.42	88.79	79.84	84.08	7.18
3	86.10	85.83	79.77	82.69	9.38
2	81.78	78.41	77.59	78.00	15.23

[1] BW – bit width; ACC – accuracy; PRE – precision; REC – recall; F1 – F1 score; FPR – false positive rate.
[2] Top models by FPR are selected from the top 10 models sorted by ACC.

(ACC $\approx$ 88–90 %, FPR $\approx$ 7–10 %). Though, the best-performed 2-bit width model lags with ACC = 83.41 % and FPR = 17.59 %. In the bottom part of Table IV, top models for each analyzed bit width selected by FPR are shown (models are selected from the top 10 models sorted by ACC). Marginal change of classification performance can be observed (1–3 %), except the model based on 32-bit width, where FPR was reduced to 2.25 % at the expense of approx. 3 % of ACC. For more insights, the dependence of model ACC on FPR (with FPR value inverted for better readability) can be seen in Figure 9. Optimal solutions, which represent a trade-off between ACC and FPR, are stressed out by Pareto fronts. Results of cloud coverage classification for best-performed 4-bit width models (4-bit width models are selected due to best accuracy/FPR ratio from quantized models) per biome using full L8-D dataset are shown in Table V. Models are selected by the highest ACC. The model performed best on the Grass/Crops biome (ACC = 95.91 % and FPR = 0.83 %). However the best FPR = 0.49 % was achieved on the Forest biome, though with low ACC = 84.01 %. The worst performance (ACC = 69.24 % and FPR = 31.11 %) was achieved on the Snow/Ice biome. Based on the results of the cloud coverage classification per biome, hypothesis is made that excluding the Snow/Ice biome (cloud coverage classification on Snow/Ice biome using natural color composite is irrelevant) from model training will improve overall model performance (especially FPR). For a better illustration of the problem, the examples of FP tiles are presented in Figure 10.

Results of cloud coverage classification using L8-D dataset without Snow/Ice biome to train, validate and test the proposed CNN are shown in Table VI. In the upper part of the table, best-performed models selected by ACC are presented. As can be noticed, in comparison with previous models trained by full L8-D dataset the classification performance was improved (ACC $\approx$ 92–95 %, FPR $\approx$ 2.9–5.7 %). In the bottom part of Table VI, top models selected by FPR are shown (models are selected from the top 10 models sorted by ACC). In case of the 32 and 2-bit width models, there is no change in performance. However, FPR for 4 and 3-bit width models is lower, whereas 4-bit model outperforms the 32-bit width one. For a better

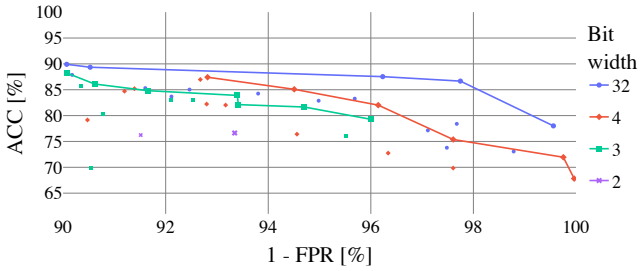


Fig. 9: Dependence of model ACC on inverse FPR value (100 % $\equiv$ 0 % FPR) using full L8-D dataset. Optimal solutions are highlighted by Pareto fronts.

illustration, the dependence of model ACC on FPR can be seen in Figure 11, where optimal solutions are highlighted by Pareto fronts.

Finally, the results of hardware architecture design space exploration are summarized. In Table VII, the overview of resource utilization measurements of quantized models using different bit widths can be found. Maximum and base folding setup was compared together with folding setup targeting 10 FPS. Even though the FPS is changing from 0.9 to 15.5, the average power consumption is stable at around 2.5 W. The parallelization settings and their respective estimated number of clock cycles for targeting specifically 10 FPS are reported in Table VIII. Results of cloud coverage classification for best-performed quantized models on FPGA can be seen in Table IX. Classification ACC and FPGA resource utilization is reported for quantized models trained using full L8-D dataset and dataset excluding the Snow/Ice biome from the dataset. The best-performed model is a quantized 4-bit width model with Snow/Ice biome excluded from training and evaluation (ACC = 94.84 %).

V. Discussion

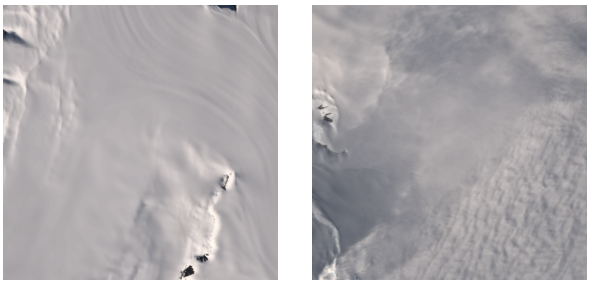
A. Quantized model for cloud detection

Based on the results of the best-performing models reported in the upper part of Table IV, the increase of the quantization level resulted in slight overall performance deterioration. Even

TABLE V: Results of cloud coverage classification for best-performed **4-bit width** models per biome employing full L8-D dataset

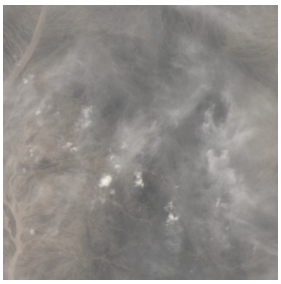
Biome	ACC [%]	PRE [%]	REC [%]	F1 [%]	FPR [%]
Barren	87.32	85.69	83.83	84.75	10.14
Forest	84.01	**99.29**	68.36	80.97	**0.49**
SnowIce	69.24	50.47	70	58.65	31.11
GrassCrops	**95.91**	98.7	**91.22**	**94.81**	0.83
Shrubland	91.73	97.03	83.14	89.55	1.89
Urban	92.89	95.01	85.23	89.86	2.62
Water	93.89	94.32	90.82	92.54	3.92
Wetlands	84.41	98.51	68.33	80.69	0.94

[1] ACC – accuracy; PRE – precision; REC – recall; F1 – F1 score; FPR – false positive rate.

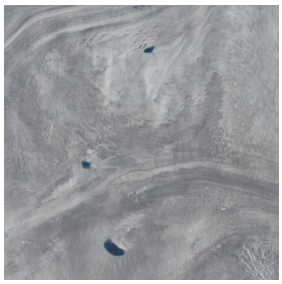


A: Snow/Ice – 0 % CCA

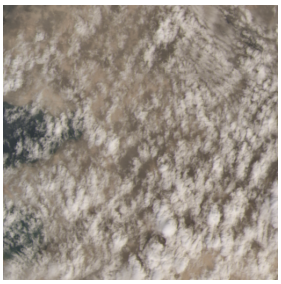
B: Snow/Ice – 42 % CCA



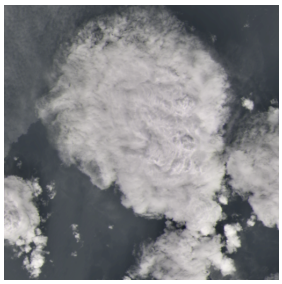
C: Barren – 10 % CCA



D: Water – 1 % CCA



E: Urban – 68 % CCA



F: Water – 58 % CCA

Fig. 10: False positive tile examples: A, B represent the Snow/Ice biome; C, D represent cloud-like FP tiles; E, F represent FP tile close to threshold (70 %).

though, the quantized models achieved comparable results to the 32-bit baseline model (except the 2-bit model). The decrease of the overall accuracy for the 4-bit and 3-bit model is just around 2 %, while for the 2-bit model it is more than 6 %. But rather than the highest overall accuracy, this study emphasizes on the low FPR (it is more convenient to process a redundant image than to discard the relevant one). Therefore, a balance between the ACC and FPR is in demand. For the baseline model and 3-bit model, the FPR is identically

TABLE VI: Results of cloud coverage classification for best-performed models using L8-D dataset excluding Snow/Ice biome.

Top models by ACC					
BW	ACC [%]	PRE [%]	REC [%]	F1 [%]	FPR [%]
32	94.92	96.12	91.93	93.98	2.81
4	94.84	92.68	95.58	94.11	5.72
3	93.37	94.24	90.13	92.14	4.17
2	92.08	92.87	88.41	90.59	5.14
Top models by FPR					
BW	ACC [%]	PRE [%]	REC [%]	F1 [%]	FPR [%]
32	94.92	96.12	91.93	93.98	2.81
4	94.30	96.82	89.72	93.14	2.23
3	92.48	94.42	87.73	90.96	3.92
2	92.08	92.87	88.41	90.59	5.14

[1] BW – bit width; ACC – accuracy; PRE – precision; REC – recall; F1 – F1 score; FPR – false positive rate.
[2] Top models by FPR are selected from top 10 models sorted by ACC.

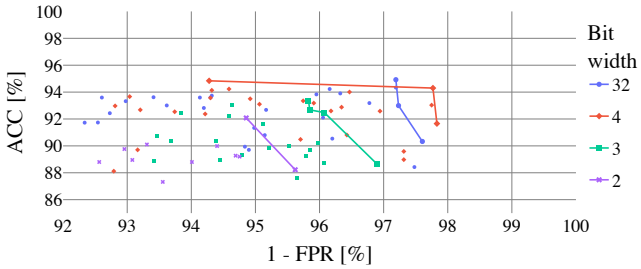


Fig. 11: Dependence of model ACC on inverse FPR value (100 % ≡ 0 % FPR) using L8-D dataset excluding Snow/Ice biome. Optimal solutions are highlighted by Pareto fronts.

TABLE VII: FPGA resource utilization and performance for different model bit widths and folding setup.

BW	Fold	FPS	APC [W]	LUT [%]	FF [%]	BRAM [%]	DSP [%]
4	max	15.468	2.592	69.05	48.61	62.50	13.64
	SPEC	9.931	2.556	66.67	48.14	61.43	13.64
	base	0.879	2.484	58.42	46.96	57.14	0.45
3	max	15.467	2.556	58.90	40.39	47.86	13.64
	SPEC	9.932	2.520	58.30	40.01	46.79	13.64
	base	0.879	2.448	55.11	39.06	42.86	0.45
2	max	15.462	2.520	47.72	32.49	32.68	13.64
	SPEC	9.927	2.484	47.74	32.29	32.50	13.64
	base	0.879	2.448	46.27	31.41	29.29	0.45

[1] BW – bit width; Fold – folding; SPEC – targeting specifically 10 FPS; FPS – frames per second; APC – average power consumption; LUT – look up table utilization; FF – flip flop utilization; BRAM – block random memory utilization; DSP – digital signal processing slice utilization

equal to 9.93 %. In the case of the 4-bit model, almost 3 % of FPR decrease can be noticed, however, the recall is lower by 10 % in comparison to the 32-bit model. The 2-bit model suffers the most from the quantization effect resulting in very insufficient FPR = 17.59 %. More balanced (ACC vs. FPR) results are provided in the bottom part of Table IV, where the best models by FPR from the top 10 models sorted by ACC are reported. Unfortunately, the performance of the quantized models keeps almost on the same levels, yet the baseline model significantly reduced its FPR to 2.25 %, while decreasing its accuracy by around 3 %. A more readable comparison of the model's performance can be seen in Figure 9. A trend

TABLE VIII: Parallelization settings and their respective estimated number of clock cycles for targeting specific 10 FPS.

layer	Specific FPS folding (PE, SIMD)	Cycles
Conv_1	(10,3)	2359296
Conv_2	(6,10)	147456
Conv_3	(3,1)	1769472
Conv_4	(1,2)	3538944
Conv_5	(1,1)	442368
Conv_6	(1,1)	331776
Conv_7	(1,1)	442368
Conv_8	(1,1)	27648
Conv_9	(1,1)	20736
Conv_10	(1,2)	442368
FC_1	(1,2)	262144
FC_2	(1,1)	1024

[1] PE – processing elements; SIMD – single instruction multiple data; Conv – convolution layer; FC – fully connected layer

TABLE IX: Results of cloud coverage classification for best-performed quantized models on FPGA.

BW	ACC (SI) [%]	ACC (EXSI) [%]	FPS	RU [%]	APC [W]
2	83.41	92.08	15.462	**31.63**	**2.520**
3	**88.24**	93.37	15.467	40.20	2.556
4	87.42	**94.84**	**15.468**	48.45	2.592

[1] BW – bit width; ACC – accuracy; SI – Snow/Ice biome included; EXSI – Snow/Ice biome excluded; FPS – frames per second; RU – FPGA resource utilization; APC – average power consumption.

of the trade-off between ACC and FPR across all quantized models together with the baseline is highlighted by Pareto fronts. It can be observed, that the baseline model outperforms the quantized ones, however, there can be found adequate alternatives to the 32-bit model.

To collect more insights and to improve the overall performance of the proposed cloud detection system, each biome of the L8-D dataset was investigated separately. We hypothesize that some biomes produce significant noise during the training process due to the false cloud-like features (snow, ice, or fog). The 4-bit models were selected to investigate biomes in quantized cases, and its results are reported in Table V. Best performance was obtained by a model trained on Grass/Crops biome with ACC $= 95.91\,\%$ and low FPR $= 0.83\,\%$. Yet, the best FPR $= 0.49\,\%$ and precision of more than $99\,\%$ was achieved by Forrest biome. However, this model lags on accuracy due to low recall $= 68.36\,\%$, which will result in a high number of undetected cloudy images. This may be caused by the cloud categories merge (thin, thick) or by the fog, which is a usual false cloud-like feature in the Forest biome [29]. Similarly, the Wetlands biome (also often affected by fog) resulted in low FPR $= 0.94\,\%$ and high precision $= 98.51\,\%$, but with low recall $= 68.33\,\%$. The Shrubland, Urban, and Water biomes achieved comparable performance with ACC from $91.73\,\%$ to $93.89\,\%$ and FPR from $1.89\,\%$ to $3.92\,\%$. The Barren biome obtained the second worse performance in terms of FPR $= 10.14\,\%$. The reason for high FPR may lie in the nature of the Barren biome, which exaggerates the thin clouds features to thick clouds. The worst performance reports the Snow/Ice biome. Low precision of $50.47\,\%$ and high FPR $= 31.11\,\%$ make its decision almost random. Since only the RGB channels were considered, the reason for misclassification is the inability to recognize between cloud, ice, and snow. To be able to classify the clouds above the snow and ice, an additional spectrum capable of altitude resolution will be necessary [6], [10], [29].

Regarding the previously mentioned results, all biomes, to a certain degree, suffer from the cloud-like features problem. An example is given in Figure 10, where six misclassified cases are presented. The first example of the Snow/Ice biome (A) has CCA $= 0\,\%$, yet the snow in the image was misclassified to a cloud. The second example of Snow/Ice biome (B) with CCA $= 42\,\%$ merged cloud with turbid snow currents. Next, the smooth hilly terrain of the Barren biome (C) stretches the features of thinly dispersed clouds. This resulted in the falsely positive image, however, the CCA is $10\,\%$ in reality. Similarly, the Water biome example (D) with CCA $= 1\,\%$ was misclassified due to the wavy, serpentine features of the shallow water. The last two examples (E, F) in Figure 10 represent the case near the threshold (CCA $= 70\,\%$). Hereabouts, a small number of cloud pixels may flip the CCA over the threshold boundary. In addition, the precise value of the CCA for each tile may be softly different from the CCA label [31], [33].

Following the reported results, the Snow/Ice biome is not suitable for the cloud detection using the proposed method. Moreover, problematic coexistence of the snow, ice and cloud in cloud detection systems has been also identified by [6], [10], [29]. Therefore, we decided to withdraw the Snow/Ice biome from the train, validation and test datasets, and to perform the experiment without this noisy data. In the real use case, the cloud detection system can omit known areas permanently covered by snow or ice from the analysis. Based on the results reported in Table VI, assumed improvements of all metrics can be observed. The best performing baseline model achieved ACC $= 94.92\,\%$ with FPR $= 2.81\,\%$. Top quantized models obtained comparable accuracy from $94.84\,\%$ to $92.02\,\%$, and FPR from $2.23\,\%$ to $5.72\,\%$. We would like to stress out, that 4-bit quantized model performed slightly better in terms of precision ($96.82\,\%$) and FPR ($2.23\,\%$) in comparison to the 32-bit model. This makes it a proper quantized substitution for deployment on FPGA. Results of this analysis confirm our hypothesis that Snow/Ice biome is naturally prone to being false positive when using RGB channels only.

In Figure 11 the accuracy vs. FPR is visualized for models trained with excluded Snow/Ice biome. From elevated position of all models within this Figure it is evident that accuracy increased all-around in comparison to Figure 9. Curves of Pareto fronts lie closer together and to the baseline front, as the quantization takes a lower toll on models performance when without visually ambiguous data.

Based on these results, following observations will be emphasised to make a deduction. Increased quantization did not cause substantial drop in values for evaluation metrics scores of results with excluded Snow/ice biome. The 4-bit model matched or overtook baseline's metric scores in accuracy and FPR. This implies equality between representational capacity of 32-bit baseline and quantized models in classifier problems that do not require high resolution for discerning discrimina-

tive features. This statement is in line with results achieved in other works [51]–[53] dealing with the quantization.

The most relevant study, CloudScout [20], [29], used hyperspectral bands for model training, resulting to 16-bit model with ACC $= 92\,\%$ and FPR $= 1\,\%$. Our proposed method outperformed this result by a 4-bit model with higher accuracy up to $3\,\%$, however with lower FPR by $1.23\,\%$. Considering, that our model used RGB bands only (without Snow/Ice biome), the presented CloudSatNet-1 method brings promising improvements in the on-board cloud detection systems. Another relevant study [21] used a larger training dataset and achieved ACC of $91\,\%$. Nevertheless, when authors deployed the model on FPGA, a significant drop of accuracy to $64\,\%$ occurred. The method introduced in this paper does not encounter a similar issue.

B. FPGA-Based Hardware Accelerator

The quantized models were deployed in three folding configurations for each bit width setting. Throughput, power consumption, and FPGA resource utilization were measured. Models with maximum folding achieved 15 FPS with input batch size of 1 and almost 20 FPS with batch size 120 which is the maximal batch size allowed to be loaded into RAM. Increase of the FPS with higher batch size was expected, and also confirmed by [3]. Power consumption measured with a USB power meter reported an increase of just over ≈ 0.2 W during the inference, compared to the resting state. In comparison with related studies, the authors of CloudScout [20], [29] reported a throughput of 2.89 FPS and 1.8 W of power consumption using Myriad VPU with 512x512x3 input size, 7 FPS and 3.4 W of power consumption using Zynq Ultrascale+ ZCU106, and 3.77 FPS using XQRKU060 solution (estimation only). Next, in the study by Reiter et al. [21], the authors reported 358.1 FPS with a much smaller input size 32x32x3, and maximum power consumption of 2.4 W. Regarding these results, the throughput and power consumption of the hardware accelerator achieved in this study is comparable with the current state-of-the-art solutions.

Based on the estimated number of cycles per layer reported in Table III, it is visible that a bottle-neck in the first layer limited the optimal throughput, and it would require a change in the network architecture to allow a higher throughput target. It was demonstrated that the network throughput can be controlled to target a specific FPS desired by the needs of the mission. A set of experiments were conducted to target specific throughput of 10 FPS. Used parallelization settings are reported in Table VIII. This approach may be useful when the instrument on the CubeSat does not have a high throughput, e. g. the camera is generating data at lower FPS. It showed flexibility in throughput control of the FPGA-based hardware accelerator created by the FINN framework. The differences for each bit width are in FPGA resource utilization, where the 2-bit model in base folding configuration utilized the lowest number of the resources (LUT = 46.27 %, FF = 31.41 %, BRAM = 29.29 %, DSP = 0.45 %). This is achieved due to no parallelization and a low memory footprint of 2-bit weights and activations. It shows the potential to reduce bit width for weights and activations even further to 1-bit and experiment with BNN in the future to enable higher throughput and deeper network on the same FPGA. As presented in Table VII, DSP slices for the first layer were selected to be utilized by Vivado just for SPEC and max folding in all bit width configurations. Memory footprint (BRAM utilization) varies from 1.43 Mb to 3,06 Mb in the ascending order relative to bit width.

VI. Conclusion

Most of the RSI is contaminated by the clouds, hence the quick and accurate method of cloud removal running on-board of the satellite has potential to significantly save the downlink. In this study, we introduced CloudSatNet-1, an FPGA-based hardware-accelerated quantized CNN for satellite on-board cloud coverage classification. We can conclude that the weights and activations quantization has a minimal or no effect on the model accuracy. However, the memory footprint reduction allows the model deployment and testing on low-cost FPGA Xilinx Zynq-7020. Using the L8-D dataset and its RGB bands only, up to $90\,\%$ of accuracy was achieved. Next, we omitted the Snow/Ice biome tiles from the dataset due to high noise production. The accuracy increased up to $94.4\,\%$ of accuracy with low FPR $= 2.23\,\%$ for the 4-bit width model. With the maximum parallelization settings, the hardware accelerator achieved 15 FPS with 2.5 W of average power consumption (0.2 W increase over the idle state). Additionally, we proved that we can control throughput to target specific FPS for the proposed classifier. Considering the reported results, the presented novel approach achieved outcome comparable with the state of the art.

The presented solution has several limitations that we would like to stress out. Firstly, the high number of false positive tiles with a terrain containing cloud-like features may be in the future compensated with the analysis involving the multispectral bands. Next, the cloud categories from the original L8-D dataset were merged to form a binary problem. Therefore, this study did not evaluate the result on the original cloud categories of the L8-D dataset, which might provide more insights on miss-classifications. Furthermore, we did not cover effects of the radiation on the cloud detection system and the redundancy will be subject of the future works. This work is the beginning of the greater effort to provide solutions based on AI for the space missions that can benefit from it, thus this work is a pilot one in nature. In the future, we aim to improve this solution to provide semantic segmentation of clouds with clouds categorization to respective classes compensating the binary decision provided in this study.

Acknowledgment

The authors would like to thank the editors and reviewers for their valuable suggestions.

References

[1] M. Lofqvist and J. Cano, "Optimizing data processing in space for object detection in satellite imagery," *arXiv preprint arXiv:2107.03774*, 2021.

[2] H. Heidt, J. Puig-Suari, A. Moore, S. Nakasuka, and R. Twiggs, "Cubesat: A new generation of picosatellite for education and industry low-cost space experimentation," in *14th Annual AIAA/USU Conference on Small Satellites*. Digital Commons, 2000.

[3] A. Maskey and M. Cho, "Cubesatnet: Ultralight convolutional neural network designed for on-orbit binary image classification on a 1u cubesat," *Engineering Applications of Artificial Intelligence*, vol. 96, p. 103952, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0952197620302700

[4] E. Kulu, "Nanosats database," 2022. [Online]. Available: https://www.nanosats.eu/

[5] EUSPA, "Euspa eo and gnss market report," 2022. [Online]. Available: https://www.euspa.europa.eu/sites/default/files/uploads/euspa_market_report_2022.pdf

[6] J. Yang, J. Guo, H. Yue, Z. Liu, H. Hu, and K. Li, "Cdnet: Cnn-based cloud detection for remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 57, no. 8, pp. 6195–6211, 2019.

[7] A. A. Fenta, H. Yasuda, N. Haregeweyn, A. S. Belay, Z. Hadush, M. A. Gebremedhin, and G. Mekonnen, "The dynamics of urban expansion and land use/land cover changes using remote sensing and spatial metrics: the case of mekelle city of northern ethiopia," *International Journal of Remote Sensing*, vol. 38, no. 14, pp. 4107–4129, 2017.

[8] M. D. King, S. Platnick, W. P. Menzel, S. A. Ackerman, and P. A. Hubanks, "Spatial and temporal distribution of clouds observed by modis onboard the terra and aqua satellites," *IEEE transactions on geoscience and remote sensing*, vol. 51, no. 7, pp. 3826–3852, 2013.

[9] Y. Zhang, W. B. Rossow, A. A. Lacis, V. Oinas, and M. I. Mishchenko, "Calculation of radiative fluxes from the surface to top of atmosphere based on isccp and other global data sets: Refinements of the radiative transfer model and the input data," *Journal of Geophysical Research: Atmospheres*, vol. 109, no. D19, 2004.

[10] J. Guo, J. Yang, H. Yue, H. Tan, C. Hou, and K. Li, "Cdnetv2: Cnn-based cloud detection for remote sensing imagery with cloud-snow coexistence," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 59, no. 1, pp. 700–713, 2020.

[11] Z. Wu, J. Li, Y. Wang, Z. Hu, and M. Molinier, "Self-attentive generative adversarial network for cloud detection in high resolution remote sensing images," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 10, pp. 1792–1796, 2019.

[12] L. Li, X. Li, L. Jiang, X. Su, and F. Chen, "A review on deep learning techniques for cloud detection methodologies and challenges," *Signal, Image and Video Processing*, vol. 15, no. 7, pp. 1527–1535, 2021.

[13] A. Kreuter, M. Zangerl, M. Schwarzmann, and M. Blumthaler, "All-sky imaging: a simple, versatile system for atmospheric research," *Applied optics*, vol. 48, no. 6, pp. 1091–1097, 2009.

[14] C. N. Long, J. M. Sabburg, J. Calbó, and D. Pagès, "Retrieving cloud characteristics from ground-based daytime color all-sky images," *Journal of Atmospheric and Oceanic Technology*, vol. 23, no. 5, pp. 633–652, 2006.

[15] D. Frantz, E. Haß, A. Uhl, J. Stoffels, and J. Hill, "Improvement of the fmask algorithm for sentinel-2 images: Separating clouds from bright surfaces based on parallax effects," *Remote sensing of environment*, vol. 215, pp. 471–481, 2018.

[16] Z. Zhu and C. E. Woodcock, "Automated cloud, cloud shadow, and snow detection in multitemporal landsat data: An algorithm designed specifically for monitoring land cover change," *Remote Sensing of Environment*, vol. 152, pp. 217–234, 2014.

[17] Y. Sui, B. He, and T. Fu, "Energy-based cloud detection in multispectral images based on the svm technique," *International Journal of Remote Sensing*, vol. 40, no. 14, pp. 5530–5543, 2019.

[18] Y. Zi, F. Xie, and Z. Jiang, "A cloud detection method for landsat 8 images based on pcanet," *Remote Sensing*, vol. 10, no. 6, p. 877, 2018.

[19] P. Miralles, A. F. Scannapieco, N. Jagadam, P. Baranwal, B. Faldu, R. Abhang, S. Bhatia, S. Bonnart, I. Bhatnagar, B. Batul, P. Prasad, H. Ortega-González, H. Joseph, H. More, S. Morchedi, A. K. Panda, M. Z. Di Fraia, D. Wischert, and D. Stepanova, "Machine learning in earth observation operations: A review," in *72nd International Astronautical Congress (IAC)*. International Astronautical Federation, October 2021.

[20] E. Rapuano, G. Meoni, T. Pacini, G. Dinelli, G. Furano, G. Giuffrida, and L. Fanucci, "An fpga-based hardware accelerator for cnns inference on board satellites: Benchmarking with myriad 2-based solution for the cloudscout case study," *Remote Sensing*, vol. 13, no. 8, p. 1518, 2021.

[21] P. Reiter, P. Karagiannakis, M. Ireland, S. Greenland, and L. Crockett, "Fpga acceleration of a quantized neural network for remote-sensed cloud detection," in *7th International Workshop on On-Board Payload Data Compression*, 2020.

[22] vloncar, "fastmachinelearning/hls4ml: coris," Nov. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5680908

[23] M. Blott, T. B. Preußer, N. J. Fraser, G. Gambardella, K. O'brien, Y. Umuroglu, M. Leeser, and K. Vissers, "Finn-r: An end-to-end deep-learning framework for fast exploration of quantized neural networks," *ACM Transactions on Reconfigurable Technology and Systems (TRETS)*, vol. 11, no. 3, pp. 1–23, 2018.

[24] 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 Sensing*, vol. 11, no. 7, p. 762, 2019.

[25] Z. Zhang, A. Iwasaki, G. Xu, and J. Song, "Cloud detection on small satellites based on lightweight u-net and image compression," *Journal of Applied Remote Sensing*, vol. 13, no. 2, p. 026502, 2019.

[26] J. Zhang, X. Li, L. Li, P. Sun, X. Su, T. Hu, and F. Chen, "Lightweight u-net for cloud detection of visible and thermal infrared remote sensing images," *Optical and Quantum Electronics*, vol. 52, no. 9, pp. 1–14, 2020.

[27] Z. Zhang, G. Xu, and J. Song, "Cubesat cloud detection based on jpeg2000 compression and deep learning," *Advances in Mechanical Engineering*, vol. 10, no. 10, p. 1687814018808178, 2018.

[28] J. H. Park, T. Inamori, R. Hamaguchi, K. Otsuki, J. E. Kim, and K. Yamaoka, "Rgb image prioritization using convolutional neural network on a microprocessor for nanosatellites," *Remote Sensing*, vol. 12, no. 23, p. 3941, 2020.

[29] G. Giuffrida, L. Diana, F. de Gioia, G. Benelli, G. Meoni, M. Donati, and L. Fanucci, "Cloudscout: a deep neural network for on-board cloud detection on hyperspectral images," *Remote Sensing*, vol. 12, no. 14, p. 2205, 2020.

[30] S. Greenland, M. Ireland, C. Kobayashi, P. Mendham, M. Post, and D. White, "Development of a minaturised forwards looking imager using deep learning for responsive operations," in *4S Symposium*. ESA, 2018.

[31] S. Foga, P. L. Scaramuzza, S. Guo, Z. Zhu, R. D. Dilley Jr, T. Beckmann, G. L. Schmidt, J. L. Dwyer, M. J. Hughes, and B. Laue, "Cloud detection algorithm comparison and validation for operational landsat data products," *Remote sensing of environment*, vol. 194, pp. 379–390, 2017.

[32] B. Markham, J. Storey, and R. Morfitt, "Landsat-8 sensor characterization and calibration," 2015.

[33] USGS, "Landsat data continuity mission level 1 (l1) data format control book (dfcb), ldcm-dfcb-004, version 6.0," 2012. [Online]. Available: https://earth.esa.int/eogateway/documents/20142/37627/LDCM-Level-1-Data-Format-Control-Book.pdf

[34] T. R. Loveland, B. C. Reed, J. F. Brown, D. O. Ohlen, Z. Zhu, L. Yang, and J. W. Merchant, "Development of a global land cover characteristics database and igbp discover from 1 km avhrr data," *International Journal of Remote Sensing*, vol. 21, no. 6-7, pp. 1303–1330, 2000. [Online]. Available: https://doi.org/10.1080/014311600210191

[35] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," *CoRR*, vol. abs/2103.13630, 2021. [Online]. Available: https://arxiv.org/abs/2103.13630

[36] A. Pappalardo, "Xilinx/brevitas," 2021. [Online]. Available: https://doi.org/10.5281/zenodo.3333552

[37] "Quantization," https://pytorch.org/docs/stable/quantization.html, accessed: 2021-11-28.

[38] M. Courbariaux, I. Hubara, D. Soudry, R. El-Yaniv, and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1," *arXiv preprint arXiv:1602.02830*, 2016. [Online]. Available: https://arxiv.org/abs/1602.02830

[39] T. Alonso, L. Petrica, M. Ruiz, J. Petri-Koenig, Y. Umuroglu, I. Stamelos, E. Koromilas, M. Blott, and K. Vissers, "Elastic-df: Scaling performance of dnn inference in fpga clouds through automatic partitioning," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 15, no. 2, dec 2021. [Online]. Available: https://doi.org/10.1145/3470567

[40] D. Przewlocka, M. Wasala, H. Szolc, K. Blachut, and T. Kryjak, "Optimisation of a siamese neural network for real-time energy efficient object tracking," *CoRR*, vol. abs/2007.00491, 2020. [Online]. Available: https://arxiv.org/abs/2007.00491

[41] N. Mellempudi, A. Kundu, D. Mudigere, D. Das, B. Kaul, and P. Dubey, "Ternary neural networks with fine-grained quantization," *CoRR*, vol. abs/1705.01462, 2017. [Online]. Available: http://arxiv.org/abs/1705.01462

[42] S. Zhou, Z. Ni, X. Zhou, H. Wen, Y. Wu, and Y. Zou, "Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *CoRR*, vol. abs/1606.06160, 2016. [Online]. Available: http://arxiv.org/abs/1606.06160

[43] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015. [Online]. Available: http://arxiv.org/abs/1502.03167

[44] Y. Umuroglu and M. Jahre, "Streamlined deployment for quantized neural networks," *CoRR*, vol. abs/1709.04060, 2017. [Online]. Available: http://arxiv.org/abs/1709.04060
[45] L. Crockett, D. Northcote, and C. Ramsay, *Exploring Zynq MPSoC: With PYNQ and Machine Learning Applications*. Strathclyde Academic Media, 2019.
[46] Xilinx, "Zynq-7000 soc data sheet: Overview," 2018.
[47] "Pynq: Python productivity," http://www.pynq.io/, accessed: 2022-02-08.
[48] O. R. developers, "Onnx runtime," https://onnxruntime.ai/, 2021, version: x.y.z.
[49] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.
[50] J. Faraone, G. Gambardella, D. Boland, N. Fraser, M. Blott, and P. H. Leong, "Customizing low-precision deep neural networks for fpgas," in *2018 28th International Conference on Field Programmable Logic and Applications (FPL)*, 2018, pp. 97–973.
[51] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding," *arXiv preprint arXiv:1510.00149*, 2015.
[52] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients," *arXiv preprint arXiv:1606.06160*, 2016.
[53] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained ternary quantization," *arXiv preprint arXiv:1612.01064*, 2016.

Radoslav Pitonak was born in Liptovsky Mikulas, Slovak republic. He earned his bachelor degree from Information technology in 2019 at the Brno University of Technology, Czech republic.

He is a Lead Software engineer at Zaitra Ltd. with focus on Machine learning and Satellite data analysis. He is participating on development of microgravity biological experiments control software on ESA project Biomission 2019. He is involved in mission definition and feasibility of data processing unit in spectrograph payload on SLAVIA mission funded by ESA. His research interests are Earth observation image analysis and deep learning for computer vision.

Jan Mucha was born in Liptovsky Mikulas, Slovak Republic. He received the Ph.D degree from data science in 2021 at the Brno University of Technology, Czech Republic.

He is a leading researcher at the Brain Diseases Analysis Laboratory, Department of Telecommunications of the Brno University of Technology and a member of the Human-Machine Interaction Group of the Signal Processing Laboratory. He is a co-author of more than 14 publications indexed by the Web of Science. He deals with the research of object detection from low contrast aerial data and with the research of novel methods based on the fractional calculus and its application in signal processing. In addition, he works on the non-invasive neurodegenerative and neurodevelopmental disorders analysis based on speech, and handwriting processing. Dr. Mucha serves as a reviewer of EU and national projects and as a reviewer of several international journals (e. g. IEEE Access, Frontiers in Psychology).

Lukas Dobis was born in Bratislava, Slovak Republic. He received the Masters degree in biomedical engineering in 2020 at the Faculty of Electrical engineering and communication at the Brno University of Technology, Czech Republic.

He is continuing his studies with Masters degree in Information Technology and Artificial Intelligence with Machine learning specialization at the Faculty of Information Technology at the Brno University of Technology. He is a machine learning enthusiast for the last three years with focus on machine learning in image processing. Currently, he is working in Zaitra Ltd. as a Machine learning engineer. In his spare time, he tries to gain experience in the fields of natural language processing, reinforcement learning and improving his linear algebra skills.

Martin Javorka was born in Liptovsky Mikulas, Slovak Republic. He received the bachelor's degree in the field of Information Technology at the Faculty of Information Technology, Brno University of Technology in 2020.

He is a co-founder of Zaitra Ltd., where he is involved in two ESA projects. In the first project Biomission 2019, the work involves automation of various liquids delivery to microgravity biological experiments. In the second project Slavia, we are working on mission definition and mission feasibility for a data processing unit in spectrograph payload. His research interest includes image analysis, deep learning, and remote sensing.

Marek Marusin received a bachelor's degree in computer science at the Faculty of Information Technology, Brno University of Technology in 2019.

He is currently a Co-founder and Chief Executive Officer at Zaitra Ltd., where he leads teams responsible for R&D activities, business development and projects related to ESA space missions. He specializes in AI solutions for Space Missions focused on Upstream Edge Intelligence. His interests include machine learning, edge computing, in-orbit data processing, and commercial space activities.