# Trading Data Size and CNN Confidence Score for Energy Efficient CPS Node Communications

Issam Raïs*, Otto Anshus*, John Markus Bjørndalen*, Daniel Balouek-Thomert†, Manish Parashar†
Department of Computer Science, UiT The Arctic University of Norway, Tromso, Norway*
Rutgers Discovery Informatics Institute, Rutgers University, Piscataway NJ, USA†
Corresponding author: issam.rais@uit.no

***Abstract*—In a context of Cyber-Physical Systems (CPS), energy-efficiency is a critical factor to achieve long operational life-time. The constraint of using battery-powered devices adds degrees of complexity, especially in a hard to reach environment with scarce network and energy resources. The reporting of data consumes large amount of energy, reducing the life-time of both individual nodes and the CPS as a whole. One way to reduce the energy cost of communication is to reduce the number of Bits to transmit. However, this is a viable approach only if the transmitted data remain suitable for further analysis.**

**In this paper, we report on the effect of reducing the image dimensions on the confidence score computed by a convolutional neural network (CNN) determining the species of animals present in images. We also report on the energy consumption of transmitting full vs. reduced dimensions of images. CPS devices and CNNs developed by the Distributed Arctic Observatory (DAO) project are used as experimental platforms.**

**The results show that the energy needed to report the images can be reduced by up to $98\%$ while only reducing the average confidence of determining the species correctly by $0.10\%$.**

***Index Terms*—CPS, machine learning, energy efficiency, tundra, monitoring, ;**

## I. Introduction

The arctic tundra is one of the most sensitive eco-system to climate change. It is a large area with presently too few large scale observation sites [1]. Scientific observatories provide on-field data to researchers in order to observe and model complex environments with rapidly changing conditions [2]–[7]. Gathering, processing and reporting of observations are often limited by the availability of sufficient energy. The reporting is also limited by the availability of a data network with sufficient bandwidth and latency. The opportunities provided by the data are consequently limited by the availability of the critical resources: energy and back-haul data networks.

One approach to implement a scientific observatory is to use a cyber-physical system (CPS) comprised of multiple devices with a set of sensors and actuators. The CPS is deployed onto an environment and used to observe its flora, fauna, atmospheric conditions and many other ecological parameters. By increasing the number of CPS devices, increasingly larger areas of the environment can be observed and manipulated. To do so, devices must be integrated to form a distributed system.

The Distributed Arctic Observatory (DAO) project at the University of Tromsø, the Arctic University of Norway, is the use case of this paper. The project develops a CPS of devices called Observation Units (OUs) for the arctic tundra. The DAO system observes the tundra and reports the observations, at near live delays. Convolution Neural Networks (CNN), an implementation of the paradigm for classification of images, are used to determine the species of animals captured.

This paper focuses on the need to reduce the energy consumption when reporting images from the devices. Lossy compression is used to reduce the size in Bits of images. This energy leverage can be achieved either by the camera itself, or by edge computing. The first approach is studied, by configuring the cameras to produce images of reduced dimensions. We document the energy saved when transmitting the reduced vs. full images. By decreasing image dimensions, the CNN will eventually produce lower confidence scores for animal species prediction. We document the impact of various image dimensions on the confidence scores. Finally, the trade-off between chosen image dimensions, confidence scores and energy savings is highlighted.

The contributions of this paper are:

- Combining a CNN and a CPS to reduce the communication costs (energy, storage and bandwidth usage) of devices taking and sending images to a trained CNN
- Reducing the energy consumption and increasing the lifetime of devices on the field while only using the dimensions of images as an energy leverage
- Documenting that a significant reduction in the communication related energy consumption through image dimensions has insignificant impact on the confidence scores
- Documenting the relationship between image size in Bits and communication related energy savings
- Applying the proposed energy leverage on a unique use case: the Distributed Arctic Observatory (DAO) project

The remaining of this paper is structured as follows. Section II presents the use case of this paper, the Distributed Arctic Observatory while section III describes the CNN application used in the DAO project to detect animals. Section IV presents the experimental setup with used metrics and simulation setup and scenarios. Sections V and VI present experimental results concerning losses for CNN confidence scores and simulated energy consumption results on various scenarios, respectively. Section VII presents the related work. Finally, Section VIII concludes this work.

## II. Motivating use-case: the DAO project

This section presents the use-case of this work: the DAO project. First, the arctic tundra and the difficulties to monitor it are covered. Then, the needs and the challenges for a distributed observatory are exposed. Finally, the current bait-cameras and our Observation Unit (OU) forming the basic block of the DAO system are presented.

### A. *The arctic tundra, a complicated eco-system*

The arctic tundra is a very large, remote, hard to reach, and potentially dangerous eco-system. By observing its flora, fauna and environmental parameters, changes can be identified and tracked. Presently, much less than 1% of the arctic tundra is monitored. However, it is the most sensitive eco-system to climate change [1]. Consequently, to accurately detect climate change, larger observations of the arctic tundra are needed.

The COAT initiative is tasked with observing the Norwegian arctic tundra, detect and explain climate related changes to advise the public and the authorities. First the state of the arctic tundra is determined based on satellite and ground-based measurements of the flora, fauna, weather, and the atmosphere to creates multiple data sets. Second, the data sets are processed to detect interesting events, like the species of animals captured in images, creating multiple new data sets. Third, the new data sets are analyzed to extract significant information, like the number of foxes and eagles detected at the different monitored sites. These insights are then used as input to climate models. Finally, based on previous results, human understanding and decision making take place [1].

Both satellite and ground-based observation platforms provide useful data to track the state of the arctic tundra. But a ground-based observation system can observe larger areas, do measurements at any time and rapidly react to local events both above and below ground, snow and ice, and do measurements at very high resolutions. Data can be reported back at any time, regularly, or on-demand. Significant processing and storage resources can be added to the devices to enable edge computing. Thus, the DAO project focuses on ground-based observation approaches.

### B. *Towards a Distributed Arctic Observatory (DAO)*

There are three major obstacles to consider when building an observation system for the arctic tundra : (i) The lack of roads and associated infrastructure implies the impossibility to realistically visit a limited number of sites in order to fetch data, supply energy, or do repairs and updates. (ii) The limited or non-existing availability of a back-haul data network for doing automated reporting of data. (iii) The lack of energy to use devices with advanced functionalities and still get a long operational lifetime.

A distributed arctic observatory system must carefully manage two fundamental resources: wireless data networks and energy. Devices are working on a limited energy budget delivered from batteries. As it is a complicated scenario, with bad weather and no long sun exposition during winter, swapping batteries by humans and regular energy harvesting are not plausible solutions. In addition, a set of functionalities are needed by the devices, including autonomous operations to save energy while still striving to observe and report.

While a back-haul network cannot be expected to be available as the common case, a device can have multiple local networks enabling communication with neighbours. Using a multi-hop approach, data can be reported through multiple units and finally to one or more units having access to back-haul networks or which are located to be reachable by humans or drones [8]. However, using the radio is energy-expensive. One approach to reduce transmission related energy consumption is to reduce the number of Bits to exchange from devices, but such leverage is only applicable if the data can be used to get close to the same analytic precision.

### C. *The bait-cameras*

COAT researchers presently use several approaches and instruments to observe the arctic tundra [9], [10]. Tens to a few hundreds of small dedicated instruments are typically deployed according to where interesting events are expected. One significant event is the presence of animals ahead of a meat bait. These bait-cameras are deployed to capture images of the animals. For hard to reach installations, it can take up to 6-12 months before humans visit the site to fetch the data, while an easy to reach installation can be reached in the span of 1-2 months.

Such stand-alone instruments with limited functionalities are based on micro-controllers. They are cheap, small and use little energy. Such characteristics make them well suited for embedded non-advanced tasks like reading a sensor and storing its value locally. However, micro-controller based instruments have none or limited network and computing resources compared to computer-based instruments making it impossible to build a complete system only relaying on them.

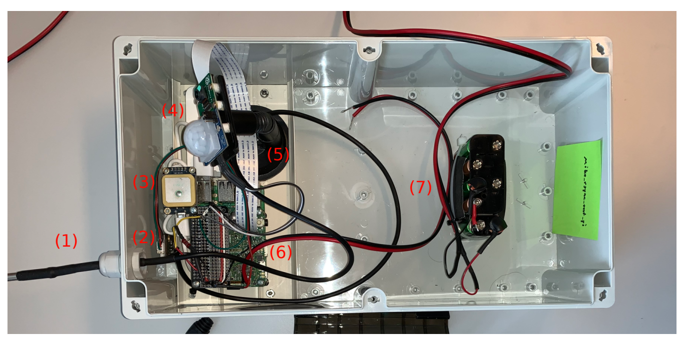



Fig. 1: Prototype Observation Unit (OU), Raspberry Pi based

### D. *The DAO Observation Unit (OU)*

For advanced functionalities, instruments with tiny computers are needed, allowing more resources and better programming support. Figure 1 introduces the basic block of the DAO system, the Observation Unit (OU). It comprises a combination of Raspberry Pi computers and Sleepy Pi micro-controller (annotated (6) in Figure 1). Sensors include optical

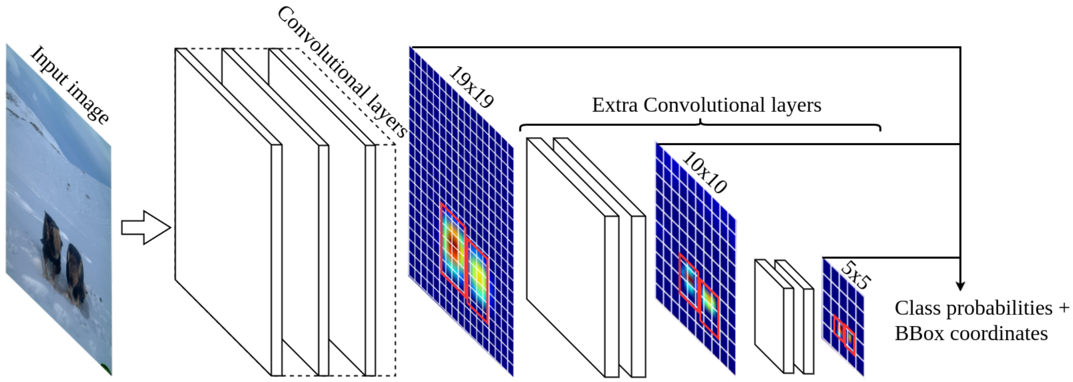



Fig. 2: Structure of the Convolutional Neural Network (CNN) designed to classify animals from the Arctic Tundra [11]

and proximity cameras (5), temperature and humidity inside (2) and outside the OU (1) and GPS (3). A range of network technologies are available from the Raspberry Pi and the 4G stick (4), including WiFi, Bluetooth, 4G LTE and LTE Cat M1. OUs are built with internal and external batteries (7). Functionalities are implemented in Go and Python.

These well-known components, platforms, languages and tools enable us to build systems relatively faster, have significant edge compute resources available, gain experience with network and energy issues, and get practical experience. We presently have 11 units deployed on the arctic tundra and around 50 units in the laboratory being used to develop functionality both for individual units and for integrating units into the DAO system.

## III. Classification of animals : description, usage

This section describes the CNN used by the DAO project to detect animal species from bait-camera images. We then describe the usage of the CNN together with OUs on the arctic tundra.

### A. *CNN for bait-camera pictures*

Convolutional Neural Networks (CNN) are commonly used to analyze and automatically annotate images. They are organized as layers, set of mathematical functions to apply on the input. The output of one layer is fed as input to the next layer. Figure 2 illustrates the layered architecture of the CNN used in this paper. For image analysis, the first layer reads the input image, while the last layer outputs the animal species predictions.

CNNs are trained by running a set of already annotated images through the CNN, comparing the output prediction against the provided ground truth and feeding adjustments back into the network. The process is repeated several times to gradually improve the CNN's ability to give correct predictions. When a limited number of annotated images are available as ground truth for training, pre-trained models are used to reduce training time and to improve the CNN output. Feature classification can be refined from a pre-trained model, rather than trained from scratch.

The CNN used for this paper is the Single Shot MultiBox Detector (SSD) model [12]. A previous contribution for the DAO project [11] investigated multiple CNN models for annotating images from the COAT bait-cameras and found the SSD to be the best performing model (measured using an evaluation metric for machine learning techniques named mean Average Precision, mAP). It outputs, for every image, a list of predictions with confidence scores. The confidence score metric reflects the confidence of the CNN that a given object in an image is an animal belonging to a certain species.

The implementation used in this paper uses the original Caffe implementation [12] as a CNN back-end, with minor changes to the Caffe Python interface for Python 3 compatibility. A pre-trained model using the ImageNet [13] dataset is used. Then, the training uses a set of 7999 annotated images from the COAT bait-cameras. An example of detection made by the used CNN is given Figure 3a, where the CNN detects a Red Fox with a confidence score of 99.99%.

### B. *Combining image dimension reduction and CNNs*

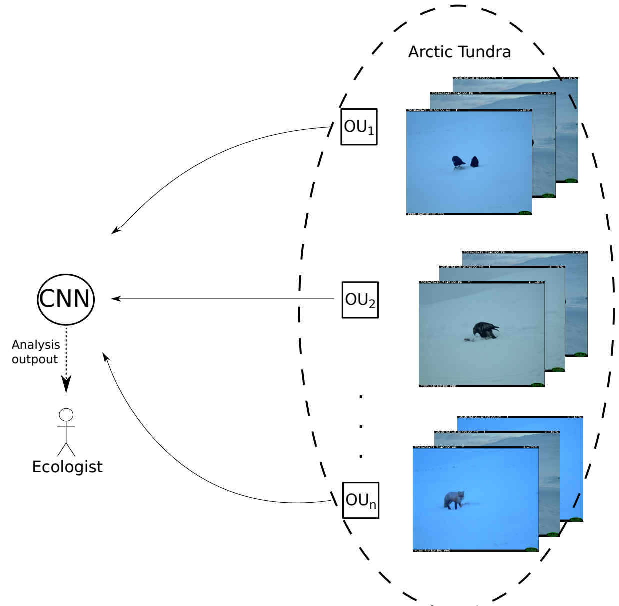
Figure 3b depicts the system used for this paper. The main actors are an ecologist as the end user, a CNN service and a set of OUs. The OUs are located at several interesting sites on the arctic tundra. The CNN receives images from the OUs deployed on the arctic tundra, and computes confidence scores with regards to which species class every found animal belongs to. The confidence scores are used by ecologists for further processing.

For this paper, we assume that the CNN application is connected to the OUs in a star topology, thus being directly accessible to all OUs.

The camera being part of the OU is pre-configured to take images at a given resolution. Images are encoded as JPEGs, stored in memory, and transmitted to the CNN. Transmission is done over a back haul network like LTE or LTE Cat M1. We assume that the CNN service is always available when an OU transmits data to it.

For this paper, we ignore where the CNN service is located along with where and when an OU should report images. We focus on the energy consumption of various OU configurations


(a) CNN detection of a Red Fox behind the meat bait, with a confidence score (cScore) of 99.99%




(b) Overview of the system, direct interactions of OUs with the CNN application

Fig. 3: CNN detection on a captured animal and overview of the system

when using the network at given bandwidths sending various amount of data.

## IV. Experimental Setup

This section presents the metrics used to evaluate the CNN confidence score with scaled down images. Then, metrics needed to evaluate the OUs to CNN communication costs are presented. Finally, a description of the simulator developed to experiment with communication related energy consumption is depicted along with simulated parameters and scenarios.

### A. CNN and energy consumption metrics

For every full image ($i$) taken by the bait-cameras without any transformations and its reduced dimensions version ($iReduced$), a set of metrics are computed: $sizeReduced$, $\Delta cScore$, $\#FP$ and $\#FN$.

$sizeReduced$ represents the factor of gain in terms of storage space when reducing the dimensions of the images.

$$sizeReduced(i) = \frac{Size(i)}{Size(iReduced)} \tag{1}$$

It is the ratio between the size in Bits of image $i$ and the size of the same image scaled down, $iReduced$.

For every image, the CNN returns a list of animal confidence scores ($cScore$), in percentage, and coordinates representing the position in the image where the CNN finds an animal. $\Delta cScore$ is the loss in terms of $cScore$ when the input to the CNN is a reduced image instead of the corresponding unmodified one.

$$\Delta cScore(i,j) = cScore(i,j) - cScore(iReduced, j) \tag{2}$$

$\Delta cScore$ is the difference of confidence scores between prediction of animal $j$ on full image $i$ and reduced image $iReduced$. A negative value for $\Delta cScore$ means that the reduced image results in a higher confidence score than the full image. For every set of images, an average $sizeReduced$ and $\Delta cScore$ along with their standard deviation are given.

$\#FP$ is the number of false positives, or the number of animals detected in the reduced images that are not in full ones. In other words, it corresponds to the number of predictions that only exist on scaled down images.

$\#FN$ is the number of false negatives, or the number of animals not detected in the reduced images that exist in full images. In other words, it corresponds to the number of predictions that exist on the full images but not on the scaled down ones.

These two metrics reveal the impact of our proposed leverage on the relevance of animal predictions. $\#FP$ and $\#FN$ are presented along with an average $cScore$ and standard deviation.

$\%eSaved$ is the percentage of energy saved by an OU concerning communication phases, when sending reduced images.

$$\%eSaved = 100 - \frac{energyComm_{Reduced} * 100}{energyComm_{Full}} \tag{3}$$

$energyComm_{Reduced}$ and $energyComm_{Full}$ represent the energy consumed during communication phases for sending the same package of reduced and full images, respectively, to the CNN service.

$tSaved$ is the up-time saved by the OU when sending reduced instead of full images.

TABLE I: Summary of simulation parameters

| | | |
|---|---|---|
| **Bandwidth** | Arctic tundra<br>USA | 0.1 to 10 Mbps<br>16.31 Mbps |
| **Data set size** | Bait-Camera<br>Stream | 1.2 GByte<br>12.0 GByte |
| **Power Consumption (Idle state)** | Rb-pi<br>Mic | 3.1W<br>0.7 W |

$$tSaved = \frac{energyComm_{Full} - energyComm_{Reduced}}{PIdle * 3600} \quad (4)$$

$PIdle$ represents the measured power of a device while being idle. $tSaved$ is the difference in energy consumed to send full vs. scaled down images expressed as time spent when the OU consumes energy at idle state. It permits to represent the impact of reducing the dimensions of the images on the saved up-time for an OU. For clarity, it is expressed in hours and minutes.

### *B. Energy and Communication Simulation*

A simulator is implemented in order to evaluate the proposed leverage in a wide variety of contexts. It explores various parameters such as bandwidth and energy consumption of nodes in a common deployment context and on the described use case, section II. This section describes the simulator and simulated infrastructure, parameters and images used.

As shown in [14], [15], the time ($T$) to transfer data ($S$ Bits) on one network link having bandwidth ($bw$) and latency as characteristics, can be computed as :

$$T = Latency + \frac{S}{bw} \quad (5)$$

The simulator assumes that the network has a constant energy consumption at both full and idle load [16], that the latency is equal to zero and that there is no congestion nor packet loss. For a device, the ideal is to send data and only consume the same amount of energy as done at idle state (which is, after shutdown state, the lowest reachable power state). Thus, the energy consumed during communication phases, $energyComm$, is simplified to the energy spent in the idle state during $T$, the time necessary to transfer the data at a given bandwidth, with no latency :

$$energyComm = T * PIdle \quad (6)$$

By ignoring the overhead of using communication capabilities of devices and latency of the network, we estimate energy savings conservatively. Like in [17], such hypothesis help simulate the best case scenarios concerning energy consumed by devices and permits to have a lower bound for the percentage of energy saved.

We only focus on the energy spent by the OUs (thus not the back-end computers nor network facilities) as we are focused on reducing the energy consumption of our CPS on the field.

The values for the simulation parameters are:

TABLE II: Simulation scenarios with corresponding bandwidth, data set size and OU idle power consumption

| **Name** | **Bandwidth** | **Data size** | **OU** |
|---|---|---|---|
| AT | ArcticTundra | Bait-Camera | Rb-pi |
| $AT_{Mic}$ | ArcticTundra | Bait-Camera | Mic |
| $AT_{Stream}$ | ArcticTundra | Stream | Rb-pi |
| $AT_{StreamMic}$ | ArcticTundra | Stream | Mic |
| USA | USA | Bait-Camera | Rb-pi |
| $USA_{Mic}$ | USA | Bait-Camera | Mic |
| $USA_{Stream}$ | USA | Stream | Rb-pi |
| $USA_{StreamMic}$ | USA | Stream | Mic |

*1) Bandwidth:* Three network scenarios for the CPS are simulated: 16, 10 and 0.1 Mbps, corresponding to a typical average bandwidth in the USA [18], good LTE and weak LTE on the arctic tundra, respectively. Last measurements are extracted from previous work from authors [8]. The last two are most relevant for the DAO use case.

*2) Data set size:* represents the amount of data that is transferred from an OU to the CNN application. Two cases are covered. The first one assumes that every OU takes images resulting in the same amount of data at every OU, 1.2 GByte, representing 3768 images from a 2018 batch of bait-camera images. Obviously, none of the images from the chosen set are used in the training set of the CNN, containing images up to year 2016. This set of images represents a deployment of approximately a month. Thus, the second one assumes that a single OU is transmitting 10 times more data, 12.0 GByte, thus representing around a year of bait camera images.

To produce the scaled down images, we convert the full images from the chosen 2018 bait-camera using the *convert* tool (from the *imagemagick* suite) and its - *scale* parameter, generating 205x154 to 1843x1382 images, corresponding to 10 to 90% of the full image dimensions, respectively. Because we control the deployed sensors, we can configure the camera. Therefore, to take picture at given dimensions, the overhead of converting images doesn't exist in our use case.

*3) Idle Power consumption:* The current generation of DAO OUs use Raspberry Pi computers. Therefore, the average idle power consumption of an OU is set to 3.1 W, corresponding to a raspberry-pi along with sensors, at the idle state. A micro-controller based OU is simulated, with a power of 0.7W [19]. A summary of the simulation parameters is given in Table I.

### *C. Simulation Scenarios*

To generate scenarios, parameters described in the previous section are combined. Such scenarios are representative of various CPS deployments. A summary of the simulation scenarios is given in Table II.

When $AT$, for Arctic Tundra, is part of the name, the scenario describes what happens if the bait-camera is sending files under an Arctic Tundra bandwidth, while $USA$ scenarios simulate a deployment under american mainland LTE coverage. When $Stream$, for stream processing, is part of the name, the scenario describes the case where the data exchanged is equal to $12GByte$. Finally, when $mic$ is part of the scenario name, the chosen idle power for the OU corresponds to the micro-controller. When nothing is stated in the scenario name for *OU* and *data exchange*, the default case implies *Rb-pi* and *Bait-camera*, respectively.

For every data set sent, the simulator takes into account the $sizeReduced$ and the relative $\Delta cScore$ under one of the previously listed scenarios to expose the possible trade-off between energy benefits and the losses of the animal classification made from images flowing from an OU being part of a CPS to a CNN application.

## V. Evaluation: CNN

In this section, the impact of reducing the image dimensions on the output of the CNN confidence score is described. Then, the different scenarios previously described are evaluated.

For each scenario, we simulate the energy consumption of the communication phases. *2048x1536 (100%)* represents the full images coming from bait cameras without any use of the leverage. The others represent a down scaling of the image done by re-configuring the camera accordingly.

### A. The impact on the confidence score

This section analyses the impact on the confidence score of reducing the dimensions of the images described in section II.

Table III presents the impact on the confidence score through several metrics. The rows represents a chosen dimension for images and the corresponding result metrics. The dimensions of the images goes from *205x154* to *1843x1382* pixels, thus respectively representing a scale down from *10%* to *90%* of the full image dimensions, *2048x1536*.

*All confidence scores* and *Confidence scores > 90%* rows represent the metrics for all the predictions and for the predictions having confidence scores over 90%, respectively. The former aids in the overall understanding of the results. The latter aids in understanding the behaviour when focusing on only the highest confidence scores. When discussing with our ecologists collaborators, confidence scores over 90% are the predictions that matches a classification done by an expert of the field. The column *#Classifications* represents the number of classifications outputted by the CNN under the given context, i.e. for a given scale and threshold of confidence score. It helps in understanding the impact of the chosen scale on the number of detected animals. The $\Delta cScore$*(StD)* column represents the average for the $\Delta cScore$ metric, and the *sizeReduced(StD)* column represents the average of the $sizeReduced$ metric. *#FP* represents the number of false positives detected from the scaled down images, and *FP avrg(StD)* represents the average confidence score of the previously detected false positives. *#FN* and *FN avrg(StD)* represent the number of false negatives and average confidence scores of detected false negatives, respectively. These metrics are previously described in section IV. Standard deviation are following the given averaged metrics, in parenthesis.

### B. "All confidence scores" analysis

To have a global view of the impact of such a leverage on the output given by the CNN, we make the analysis of chosen metrics on all outputted animal predictions, for all image scales. For *All confidence scores* and for all scales, the average difference between predictions, $\Delta cScore$ metric, is very low. For instance, for the *"10%"* image scale, $\Delta cScore$ has an average of $-1.53$ and a standard deviation of $21.75$. In other words, the average difference in confidence scores when using full vs. scaled down images to only *10%* of the full image dimensions, is around $-1.53$ per cent but with a standard deviation of $21.75$.

Note that a negative average $\Delta cScore$ means that, in average, the confidence scores are better than the full ones, here by an average of $1.53$ but with a standard deviation of $21.75$. These results show that for the smallest studied input to the CNN, the *"10%"* images, most of the predictions are deviated by, at worst, around $20\%$ when compared to confidence scores on full images.

For image scale between *"20%"* and *"90%"*, $\Delta cScore$ has a near null average and low standard deviation between $-0.98(12.97)$ and $-0.52(5.65)$, respectively. These results show that for the other studied dimensions, from "$20\%$" to *"90%"*, most of the predictions are deviated by, at worst, around $12\%$ to $5\%$ when compared to predictions using full images, respectively.

We also observe that images reduced from *"10%"* to *"90%"* of the originals makes the image size in Bits smaller by a factor of $48.79$ to $1.45$, respectively. For images reduced to *"20%"*, 13174 out of 13853 classifications are false positives. For the images reduced to 90%, 443 out of 1260 classifications are false positives.

Thus, between $95.09\%$ and $35.15\%$ of the predictions from scaled down images are false positives. But these predictions have very low average confidence score and very low standard deviation, between $1.47(4.63)$ for *205x154 (10%)* and $0.68(0.77)$ for *"90%"*, respectively. As ecologists are only interested in very high confidence scores matching what a human specialist can achieve, these false positives are not of interest to the ecologists and will not be studied.

For image dimensions *"10%"* and *"90%"*, 684 and 111 out of 912 are detected as false negatives. In other words, $75.0\%$ to $12\%$ of classifications found on the *"100%"* images are not found on *"10%"* and *"90%"*, respectively. Again, very low average $cScore$ are found, from $3.55(10.95)$ for *"10%"* to $0.97(0.82)$ for *"90%"*.

### C. "Confidence scores > 90%" analysis

To focus on the impact reducing the image dimensions has on the high confidence scores, we make the same analysis as above but focus on confidence scores of animals above 90%. The high confidence score classifications are those being of interest to ecologists because they match the classifications they can do manually, i.e matching the classification of an expert. Obviously, the number of classifications drops compared to the *All confidence scores*. For all cases, between 94 and 99 classifications match.

TABLE III: CNN confidence score differences, sizeReduced, false positives and false negatives

| Dimensions (scale) | #Classifications | $\Delta cScore$(StD) | sizeReduced(StD) | #FP | FP avrg(StD) | #FN | FN avrg(StD) |
|---|---|---|---|---|---|---|---|
| *All confidence scores* | | | | | | | |
| 205x154 (10%) | 2167 | -1.53(21.75) | 48.79(7.08) | 1910 | 1.47(4.63) | 684 | 3.55 (10.95) |
| 410x307 (20%) | 13853 | -0.98(12.97) | 19.15(2.67) | 13174 | 0.78(0.92) | 405 | 2.07 (3.99) |
| 614x461 (30%) | 8057 | 0.10 (8.75) | 10.17(1.57) | 7282 | 0.69(0.75) | 298 | 1.44 (2.10) |
| 819x614 (40%) | 4055 | -0.23(7.37) | 6.28 (0.98) | 3204 | 0.66(0.59) | 211 | 1.45 (2.31) |
| 1024x768 (50%) | 1709 | 0.01 (5.70) | 4.21 (0.67) | 909 | 0.67(0.92) | 182 | 1.07 (0.89) |
| 1229x922 (60%) | 1719 | -0.21(6.37) | 3.05 (0.41) | 908 | 0.70(1.72) | 183 | 1.24 (1.51) |
| 1434x1075 (70%) | 1778 | -0.04(5.00) | 2.29 (0.26) | 958 | 0.60(0.23) | 148 | 1.12 (1.16) |
| 1638x1229 (80%) | 1868 | -0.12(4.66) | 1.81 (0.18) | 1013 | 0.62(0.50) | 110 | 1.03 (0.86) |
| 1843x1382 (90%) | 1260 | -0.52(5.65) | 1.45 (0.12) | 443 | 0.68(0.77) | 111 | 0.97 (0.82) |
| 2048x1536 (100%) | 912 | 0.0 (0.0) | 0.0 | 0 | 0 (0.0) | 0 | 0 (0.0) |
| *Confidence scores > 90%* | | | | | | | |
| 205x154 (10%) | 94 | 8.79(21.15) | 46.89 (10.45) | 1 | 93.82(0.0) | 5 | 97.17(3.32) |
| 410x307 (20%) | 99 | 1.67(10.01) | 18.55 (3.58) | 0 | 0.0 (0.0) | 0 | 0.0 (0.0) |
| 614x461 (30%) | 99 | 1.43 (8.03) | 10.14 (1.72) | 0 | 0.0 (0.0) | 0 | 0.0 (0.0) |
| 819x614 (40%) | 99 | 0.83 (5.23) | 6.42 (0.96) | 0 | 0.0 (0.0) | 0 | 0.0 (0.0) |
| 1024x768 (50%) | 99 | 0.52 (3.42) | 4.42 (0.64) | 0 | 0.0 (0.0) | 0 | 0.0 (0.0) |
| 1229x922 (60%) | 99 | 0.41 (2.78) | 3.19 (0.39) | 0 | 0.0 (0.0) | 0 | 0.0 (0.0) |
| 1434x1075 (70%) | 99 | 0.35 (2.53) | 2.40 (0.25) | 0 | 0 (0.0) | 0 | 0 (0.0) |
| 1638x1229 (80%) | 99 | 0.18 (1.28) | 1.89 (0.18) | 0 | 0 (0.0) | 0 | 0 (0.0) |
| 1843x1382 (90%) | 99 | 0.18 (1.31) | 1.51 (0.12) | 0 | 0 (0.0) | 0 | 0 (0.0) |
| 2048x1536 (100%) | 99 | 0.0 (0.0) | 0.0 | 0 | 0 (0.0) | 0 | 0 (0.0) |

$\Delta cScore$ varies between $8.79(21.15)$ and $0.18(1.28)$. With image scales over *"40%"*, the average and the standard deviation also start to be very low, $0.83(5.23)$.

The average factor for the decrease in size is very close to *All confidence scores*, with a highest $sizeReduced$ of $46.89(10.45)$ to $1.51(0.12)$ for *"10%"* to *"90%"*, respectively. In other words, scaled down images where the CNN gives a confidence score over 90% are in average between $46.89$ and $1.51$ times smaller than full *"100%"* images, respectively.

For images scaled down to *"10%"*, $1$ out of $94$ is detected as false positives. For images with scales above *"20%"*, all with 99 predictions like for full *"100%"* images, there are zero false positives. Thus, only one prediction is detected as false positives for all studied image scales.

The CNN output false negatives only for the *"10%"* images, at $5$ false negatives. In other words, for predictions with confidence score over $90\%$ and only on *"10%"* scale images, the CNN doesn't detect $5.05\%$ of the predictions seen on full *"100%"* images. In summary, if looking only at the predictions with high confidence scores, the number of false positives and negatives for images above *"10%"* as scale is equal to zero.

## VI. Evaluation: Up-time and Energy

This section focuses on the energy benefits coming from reducing the image dimensions for communication periods for the scenarios summed up in table II. The simulator uses $sizeReduced$ factors in the *All confidence scores* in table III.

Table IV presents results of the simulation of energy consumption during communication phases for the scenarios described in Table II. For every scenario, a set of results are given: the energy consumption (in Joules (J)), the reduction in transmission time $tSaved$ (in hours) and the percentage of energy saved $\%eSaved$, achieved when reducing image dimensions.

As the bandwidth is highly variable on the Arctic Tundra, two extreme bandwidths reported in [8] are simulated under "*Arctic Tundra 0.1 Mbps Bandwidth*" and "*Arctic Tundra 10.0 Mbps Bandwidth*". A bandwidth equivalent to the average bandwidth seen in the USA is also simulate with "*USA, 16.31 Mbps as Bandwidth*", to include a typical available bandwidth.

### A. Reduced Up-time

For the low bandwidth on the Arctic Tundra, $0.1$ Mbps, major up-time savings comes from transmitting reduced images. This is the case on both Raspberry Pi ($AT$) and micro-controller based OUs ($AT_{Mic}$), for both the *Bait-Camera* and *Stream* sizes of exchanges. The reduced time to transmit are from 26h07m to 8h17m and from 261h12m to 82h46m for the *Bait-Camera* and the *Stream* scenarios, respectively.

Optimistically assuming a bandwidth of 10 Mbps on the Arctic Tundra, the energy savings result in up-time savings between 16m to 5m and between 2h37m and 50m, for the *Bait-Camera* and the *Stream* scenarios, respectively.

If the OU is deployed in a region where the coverage permits a bandwidth equal to 16.31Mbps, the up-time gains would be between 10m and 3m and between 1h36m and 30m, for *Bait-Camera* and *Stream* sizes of exchanges, respectively.

Thus, in every studied case, devices using the image scale leverage see their transmission time reduced significantly, obviously translating into energy savings.

### B. Energy saved

As expected, for every scenario, choosing a smaller scale of image consumes less energy to communicate than a bigger one. For every chosen bandwidth and size of exchange, the energy consumed (in Joules) during communication periods are an order of magnitude higher when comparing Raspberry-pi to micro controller based OUs. Also, the energy consumed while having "Stream" size of exchange is an order of magnitude higher than "Bait-Camera".

TABLE IV: Energy spent (Joules) and time saved (hours) for communication phases, one OU for all scenarios

| | | *Size : Bait-Camera (1.2 GByte)* | | | *Size : Stream (12.0 GByte)* | | | |
|---|---|---|---|---|---|---|---|---|
| *Arctic Tundra, 0.1 Mbps Bandwidth* | | | | | | | | |
| Dimensions (scale) | sizeReduced | AT(J) | $AT_{Mic}$(J) | tSaved | $AT_{Stream}$(J) | $AT_{StreamMic}$ (J) | tSaved | % eSaved |
| 205x154 (10%) | 48.79 | 6099.61 | 1377.33 | 26h07m | 60996.11 | 13773.31 | 261h12m | 97.95 |
| 410x307 (20%) | 19.15 | 15540.47 | 3509.14 | 25h16m | 155404.70 | 35091.38 | 252h44m | 94.78 |
| 614x461 (30%) | 10.17 | 29262.54 | 6607.67 | 24h03m | 292625.39 | 66076.70 | 240h27m | 90.17 |
| 819x614 (40%) | 6.28 | 47388.54 | 10700.64 | 22h25m | 473885.35 | 107006.37 | 224h12m | 84.08 |
| 1024x768 (50%) | 4.21 | 70688.84 | 15962.00 | 20h19m | 706888.36 | 159619.95 | 203h20m | 76.25 |
| 1229x922 (60%) | 3.05 | 97573.77 | 22032.79 | 17h55m | 975737.70 | 220327.87 | 179h14m | 67.21 |
| 1434x1075 (70%) | 2.29 | 129956.33 | 29344.98 | 15h01m | 1299563.32 | 293449.78 | 150h13m | 56.33 |
| 1638x1229 (80%) | 1.81 | 164419.89 | 37127.07 | 11h56m | 1644198.90 | 371270.72 | 119h20m | 44.75 |
| 1843x1382 (90%) | 1.45 | 205241.38 | 46344.83 | 8h17m | 2052413.79 | 463448.28 | 82h46m | 31.03 |
| 2048x1536 (100%) | 0.00 | 297600.00 | 67200.00 | 0h0m | 2976000.00 | 672000.00 | 0h0m | 0.0 |
| *Arctic tundra, 10 Mbps Bandwidth* | | | | | | | | |
| Dimensions (scale) | sizeReduced | AT(J) | $AT_{Mic}$(J) | tSaved | $AT_{Stream}$(J) | $AT_{StreamMic}$ (J) | tSaved | % eSaved |
| 205x154 (10%) | 48.79 | 61.00 | 13.77 | 0h16m | 609.96 | 137.73 | 2h37m | 97.95 |
| 410x307 (20%) | 19.15 | 155.40 | 35.09 | 0h15m | 1554.05 | 350.91 | 2h32m | 94.78 |
| 614x461 (30%) | 10.17 | 292.63 | 66.08 | 0h14m | 2926.25 | 660.77 | 2h24m | 90.17 |
| 819x614 (40%) | 6.28 | 473.89 | 107.01 | 0h13m | 4738.85 | 1070.06 | 2h15m | 84.08 |
| 1024x768 (50%) | 4.21 | 706.89 | 159.62 | 0h12m | 7068.88 | 1596.20 | 2h02m | 76.25 |
| 1229x922 (60%) | 3.05 | 975.74 | 220.33 | 0h11m | 9757.38 | 2203.28 | 1h48m | 67.21 |
| 1434x1075 (70%) | 2.29 | 1299.56 | 293.45 | 0h9m | 12995.63 | 2934.50 | 1h30m | 56.33 |
| 1638x1229 (80%) | 1.81 | 1644.20 | 371.27 | 0h7m | 16441.99 | 3712.71 | 1h12m | 44.75 |
| 1843x1382 (90%) | 1.45 | 2052.41 | 463.45 | 0h5m | 20524.14 | 4634.48 | 0h50m | 31.03 |
| 2048x1536 (100%) | 0.00 | 2976.00 | 672.0 | 0h0m | 29760.00 | 6720.00 | 0h0m | 0.0 |
| *USA, 16.31 Mbps Bandwidth* | | | | | | | | |
| Dimensions (scale) | sizeReduced | USA(J) | $USA_{Mic}$(J) | tSaved | $USA_{Stream}$(J) | $USA_{StreamMic}$(J) | tSaved | % eSaved |
| 205x154 (10%) | 48.79 | 37.40 | 8.44 | 0h10m | 373.98 | 84.45 | 1h36m | 97.95 |
| 410x307 (20%) | 19.15 | 95.28 | 21.52 | 0h9m | 952.82 | 215.15 | 1h33m | 94.78 |
| 614x461 (30%) | 10.17 | 179.41 | 40.51 | 0h9m | 1794.15 | 405.13 | 1h28m | 90.17 |
| 819x614 (40%) | 6.28 | 290.55 | 65.61 | 0h8m | 2905.49 | 656.08 | 1h22m | 84.08 |
| 1024x768 (50%) | 4.21 | 433.41 | 97.87 | 0h7m | 4334.08 | 978.66 | 1h15m | 76.25 |
| 1229x922 (60%) | 3.05 | 598.25 | 135.09 | 0h7m | 5982.45 | 1350.87 | 1h06m | 67.21 |
| 1434x1075 (70%) | 2.29 | 796.79 | 179.92 | 0h6m | 7967.89 | 1799.20 | 0h55m | 56.33 |
| 1638x1229 (80%) | 1.81 | 1008.09 | 227.63 | 0h4m | 10080.93 | 2276.34 | 0h44m | 44.75 |
| 1843x1382 (90%) | 1.45 | 1258.38 | 284.15 | 0h3m | 12583.78 | 2841.50 | 0h30m | 31.03 |
| 2048x1536 (100%) | 0.00 | 1824.65 | 412.01 | 0h0m | 18246.47 | 4120.17 | 0h0m | 0.0 |

Using the image scale leverage results in significant energy savings, $\%eSaved$, for all scenarios between $97.95\%$ and $31.03\%$ for *"10%"* and *"90%"* image scales, respectively. Thus, even with the largest applied image scale, we still document energy savings. As noticed, $\%eSaved$ is the same for all scenarios for a chosen percentage. We provide a proof below that the percentage of energy saved depends only on the $sizeReduced$ factor.

Let $energyComm_{Reduced}$, for energy Communication Reduced, be the energy consumed for transmitting reduced data, $energyComm_{Full}$, for energy Communication Full, be the energy needed to send full data over a given bandwidth $bw$, $Pidle$ be the idle power consumption of a given device, and $sizeO$ be the size of the full images.

**Theorem 1.** *The percentage of energy saved, for the image scale leverage, depends only on "sizeReduced".*

*Proof of Theorem 1.*

$$\begin{aligned}
\%eSaved &= 100 - \frac{energyComm_{Reduced} * 100}{energyComm_{Full}} \\
&= 100 - \frac{PIdle * \frac{\frac{sizeO}{sizeReduced}}{bw}}{PIdle * \frac{sizeO}{bw}} * 100 \\
&= 100 - \frac{\cancel{PIdle} * \frac{\frac{sizeO}{sizeReduced}}{bw}}{\cancel{PIdle} * \frac{sizeO}{bw}} * 100 \\
&= 100 - \frac{\frac{sizeO}{sizeReduced}}{bw} * \frac{bw}{sizeO} * 100 \\
&= 100 - \frac{\frac{sizeO}{sizeReduced}}{\cancel{bw}} * \frac{\cancel{bw}}{sizeO} * 100 \\
&= 100 - \frac{sizeO}{sizeReduced} * \frac{1}{sizeO} * 100 \\
&= 100 - \frac{\cancel{sizeO}}{sizeReduced} * \frac{1}{\cancel{sizeO}} * 100 \\
&= 100 - \frac{1}{sizeReduced} * 100
\end{aligned}$$

□

As $\%eSaved$ depends only on $sizeReduced$, the need to complicate the OUs design to apply this leverage is near null. In fact, the percentage of energy saved does not depend on $PIdle$, the consumption of the OU, nor $bw$, the bandwidth of the network. In other words, the OUs can compute the potential energy saved to the full images by knowing the sizes in Bits of both full and reduced image to compute $sizeReduced$. Thus, no need for monitoring the energy consumed nor the bandwidth to know the $\%eSaved$, complicating the OU and potentially reducing its up-time.

Combining previous results (from the CNN losses and energy saved for *All confidence scores*), the average difference concerning confidence scores ($\Delta cScore$) range from $-1.53$ to $0.10$ while the percentage of energy saved (*% eSaved*) range from $97.95\%$ to $31.03\%$. Thus, even when *"90%"* is chosen as the image scale, significant energy saving are made, here $31.03\%$, compared to sending full images. Importantly, this comes with almost unchanged confidence scores as both average and standard deviation of $\Delta cScore$ are low.

When looking at the *Confidence scores > 90%* and for the same image scale, $\Delta cScore$ average and standard deviation are equal to $0.18(1.31)$ with no false positives ($\#FP$) and no false negatives($\#FN$) detected. When *"50%"* is the chosen image scale, the average and standard deviation of $\Delta cScore$ are low and equal to $0.01(5.70)$ and $0.52(3.42)$, for both *All confidence scores* and *Confidence scores > 90%* respectively. No relevant false positives nor false negatives are detected and the percentage of energy saved is equal to $76.25\%$, compared to sending *"100%"* images.

These results are focused on only one OU, connected directly with the CNN application at the back-end. With every OU applying this leverage, the lifetime of the CPS will be greatly extended. For the Arctic Tundra use-case, good coverage is not expected, forcing most of the OUs to exchange data in a peer-to-peer manner. Thus, the energy savings will even be bigger when this leverage is applied.

In this section, we documented the losses concerning the confidence score, false positive and negatives of the CNN when using the proposed leverage. We underline the significant energy savings during communication periods on all the scenarios and image scales, with bait-camera and stream data sizes, at high and low bandwidth, and with OUs based on Raspberry Pis and micro-controllers.

We underline a trade-off between the losses in the confidence scores given by the CNN and the energy saved by the deployed CPS. This trade-off could be used to involve the end user, here the ecologist, in the choice of dimensions of images. Ecologists, as end users, are the only one able to tell the system how far the confidence scores can differ from the *"100%"* confidence scores.

## VII. Related work

Machine learning applications are nowadays widely used to automate multiple tasks. Convolutional Neural Network (CNN) is a paradigm used mainly for image classification. Ecologists deploy multiple occurrence of wildlife cameras to take images of wild animals in remote places, resulting in millions of images to analyze, usually by hand. Thus, CNNs are a good match for analysis of images from wildlife cameras. For example, in [20] authors implement and evaluate a ResNet-18 architecture based CNN for wildlife cameras. In [21], authors also implement a CNN but reduce the training time and better the accuracy by using transfer learning. Both previously exposed article stress the reduction of time for the analysis task for the ecologist end user.

The high computational complexity, and thus high need for energy, for both the training and usage of these models is one of the main drawback of algorithms composing this paradigm. Recent literature propose to tackle this problem.

On the hardware side, versatile approaches can be found. In [22] authors proposes a pipe-lined approach that uses multiple FPGA for better performance and energy efficiency. Authors also validate their proposed architecture by building a prototype of six connected FPGA boards to demonstrate the energy efficiency of the approach, compared to CPU and GPU approaches. In [23], authors propose a system-on-chip approach that answers the analytics of data near sensors.

There is also a wide variety of solutions on the software side. In [24] authors propose a data-flow to reduce the data movements and thus energy consumption of CNN application on physical architectures. In [25], an energy-aware algorithm that uses the energy consumed by a CNN to chose which layer to prune in order to increase its energy efficiency is proposed. To prune a layer, the algorithm removes the weights that have the smallest joint impact on the output feature maps.

A set of papers proposes to use a CNN with a CPS. In [26], authors combine a CPS, Big Data analytics and optimization algorithms for energy efficient machining optimization. For acoustic event detection in [27], reduction of the energy consumption of analytics is done by implementing a lightweight data footprint algorithm for both energy efficiency and compatibility for low power architectures, like edge devices. Finally, papers like [28] proposes to reduce the energy consumption of devices by modeling their energy consumption with learning approaches, answering parameters like the voltage or frequency for a given workload.

The introduction of machine learning solutions to automate analytics processes still has lots of open problems [29]. It would be interesting to apply all previous solutions (or at least compatible ones) to maximize energy efficiency as an end to end problem, from the CNN training to the data coming back from the CNN. But none of the previous solutions proposes to reduce the energy consumed of a CPS on batteries with modification of the output of sensors, without deteriorating the confidence score and without modifying the CNN. In other words and on top of our knowledge no trade-off between energy consumed by devices on the edge and a CNN confidence score is proposed in the literature.

## VIII. Conclusion

In a world where connected devices working from batteries are flourishing everywhere around us, reducing the energy consumed during communication periods is crucial. We propose an energy leverage to reduce the energy consumption of communication periods of units sending images to a Convolution Neural Network (CNN) application, while maintaining close to equal confidence scores, especially for high confidence value predictions. We apply this method on a real use case of Observation Units (OUs) deployed on the arctic tundra, the

Distributed Arctic Tundra (DAO) project, taking images of wild animals passing by the device.

Evaluation highlights a wide variety of energy savings thanks to scaling down the image dimensions, from 31 to 98 percent of energy saved during communication phases when devices on the field send images to a centralized CNN application. The worst case average difference concerning the confidence scores of animals on a complete set of previously collected images is a negligible $0.01$.

The lifetime of the CPS deployed on the field is increased thanks to trading off confidence scores for energy savings. Depending on the realistically chosen bandwidth and measured device characteristics, the time an observation unit must be operating just to transmit the set of images is reduced by at best 261 hours to at worst 3 minutes, underlining the fact that all the studied scenarios, including bandwidth, idle power consumption and size of exchange as parameters, could benefit from using this energy leverage.

A short term future work is a full implementation of this leverage on the DAO project to further involve the ecologists in the end choice of the image scale, as a dynamic parameter of the CPS. Future works also include exploration of other approaches to implement lossy compression on images along with data types like sensor logs, video, audio, generalizing the approach to other types of data and other automated analysis tools. Finally, we plan on implementing a lightweight edge computing version of the classification tool.

## Acknowledgment

The DAO project is supported by the Research Council of Norway (RCN) IKTPluss program, project number 270672. Thank you very much to the COAT ecologists, UiT.

## References

[1] R. A. Ims, J. U. Jepsen, A. Stien, and N. G. Yoccoz, "Science plan for coat: climate-ecological observatory for arctic tundra," *Fram Centre report series*, vol. 1, pp. 1–177, 2013.

[2] Laser Interferometer Gravitational-wave Observatory (LIGO), "https://www.ligo.caltech.edu," Last accessed on March 2019.

[3] W. Gressler, D. J., E. Hileman, D. R. Neill *et al.*, "LSST Telescope and site status," *Ground-based and Airborne Telescopes V, 91451A*, 2014.

[4] L. Evans and P. Bryant, "LHC machine," *Journal of Instrumentation*, vol. 3, 2008.

[5] B. Wang, X. Zhu, C. Gao, Y. Bai *et al.*, "Square Kilometre Array telescope - Precision reference frequency synchronisation via 1f-2f dissemination," *Scientific reports*, vol. 5, 2015.

[6] L. M. Smith *et al.*, "The Ocean Observatories Initiative," *Oceanography*, vol. 31, no. 1, 2018.

[7] The National Ecological Observatory Network (NEON), "http://www.neonscience.org," Last accessed on March 2019.

[8] I. Raïs, J. M. Bjørndalen, P. H. Ha, K.-A. Jensen, L. S. Michalik, H. Mjøen, Ø. Tveito, and O. Anshus, "Uavs as a leverage to provide energy and network for cyber-physical observation units on the arctic tundra," in *2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, 2019, pp. 625–632.

[9] P. H. HAa, R. A. IMSb, J. U. JEPSEN, S. T. KILLENGREEN, E. F. KLEIVEN, E. M. SOININEN, N. G. YOCCOZ, and A. HORSCH, "Building a sensor system for a large scale arctic observatory," *Communicating Process Architectures 2015 & 2016: WoTUG-37 & WoTUG-38*, vol. 69, p. 445, 2018.

[10] E. M. Soininen, I. Jensvoll, S. T. Killengreen, and R. A. Ims, "Under the snow: a new camera trap opens the white box of subnivean ecology," *Remote Sensing in Ecology and Conservation*, vol. 1, no. 1, pp. 29–38, 2015.

[11] H. Thom, "Unified detection system for automatic, real-time, accurate animal detection in camera trap images from the arctic tundra," Master's Thesis, UiT The Arctic University of Norway, Thom, Håvard, 2017.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 21–37.

[13] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 248–255.

[14] H. Casanova and L. Marchal, "A network model for simulation of grid application," 2002.

[15] P. Velho and A. Legrand, "Accuracy study and improvement of network simulation in the simgrid framework," in *Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. ICST (Institute for Computer Sciences, Social-Informatics and ..., 2009, p. 13.

[16] T. T. Ye, L. Benini, and G. De Micheli, "Analysis of power consumption on switch fabrics in network routers," in *Proceedings 2002 Design Automation Conference (IEEE Cat. No. 02CH37324)*. IEEE, 2002, pp. 524–529.

[17] I. Raïs, D. Balouek-Thomert, A.-C. Orgerie, L. Lefèvre, and M. Parashar, "Leveraging energy-efficient non-lossy compression for data-intensive applications," in *HPCS 2019 - 17th International Conference on High Performance Computing & Simulation*, ser. HPCS 2019 - 17th International Conference on High Performance Computing & Simulation, Dublin, Ireland, Jul. 2019, pp. 1–7. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02179621

[18] opensignal, "The state of lte (feb 2018)," https://www.opensignal.com/reports/2018/02/state-of-lte, 2018.

[19] F. Ingelrest, G. Barrenetxea, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "Sensorscope: Application-specific sensor network for environmental monitoring," *ACM Transactions on Sensor Networks (TOSN)*, vol. 6, no. 2, p. 17, 2010.

[20] M. A. Tabak, M. S. Norouzzadeh, D. W. Wolfson, S. J. Sweeney, K. C. VerCauteren, N. P. Snow, J. M. Halseth, P. A. Di Salvo, J. S. Lewis, M. D. White *et al.*, "Machine learning to classify animal species in camera trap images: applications in ecology," *Methods in Ecology and Evolution*, vol. 10, no. 4, pp. 585–590, 2019.

[21] M. Willi, R. T. Pitman, A. W. Cardoso, C. Locke, A. Swanson, A. Boyer, M. Veldthuis, and L. Fortson, "Identifying animal species in camera trap images using deep learning and citizen science," *Methods in Ecology and Evolution*, vol. 10, no. 1, pp. 80–91, 2019.

[22] C. Zhang, D. Wu, J. Sun, G. Sun, G. Luo, and J. Cong, "Energy-efficient cnn implementation on a deeply pipelined fpga cluster," in *Proceedings of the 2016 International Symposium on Low Power Electronics and Design*. ACM, 2016, pp. 326–331.

[23] F. Conti, R. Schilling, P. D. Schiavone, A. Pullini, D. Rossi, F. K. Gürkaynak, M. Muehlberghuber, M. Gautschi, I. Loi, G. Haugou *et al.*, "An iot endpoint system-on-chip for secure and energy-efficient near-sensor analytics," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 64, no. 9, pp. 2481–2494, 2017.

[24] Y.-H. Chen, J. Emer, and V. Sze, "Eyeriss: A spatial architecture for energy-efficient dataflow for convolutional neural networks," in *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3. IEEE Press, 2016, pp. 367–379.

[25] T.-J. Yang, Y.-H. Chen, and V. Sze, "Designing energy-efficient convolutional neural networks using energy-aware pruning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5687–5695.

[26] Y. Liang, X. Lu, W. Li, and S. Wang, "Cyber physical system and big data enabled energy efficient machining optimisation," *Journal of cleaner Production*, vol. 187, pp. 46–62, 2018.

[27] M. Meyer, L. Cavigelli, and L. Thiele, "Efficient convolutional neural network for audio event detection," *arXiv preprint arXiv:1709.09888*, 2017.

[28] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, S. U. Khan, and P. Li, "A double deep q-learning model for energy-efficient edge scheduling," *IEEE Transactions on Services Computing*, 2018.

[29] M. Shafique, R. Hafiz, M. U. Javed, S. Abbas, L. Sekanina, Z. Vasicek, and V. Mrazek, "Adaptive and energy-efficient architectures for machine learning: Challenges, opportunities, and research roadmap," in *2017 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. IEEE, 2017, pp. 627–632.