SSC18-WKVII-03

Machine-Learning Space Applications on SmallSat Platforms with TensorFlow

Jacob Manning, David Langerman, Barath Ramesh, Evan Gretok, Christopher Wilson, Alan George
NSF Center for Space, High-performance, and Resilient Computing (SHREC)
4420 Bayard Street, Suite #560, Pittsburgh, PA, 15213; 412-383-8122
jacob.manning@chrec.org

James MacKinnon, Gary Crum
NASA Goddard Space Flight Center
8800 Greenbelt Rd, Greenbelt, MD, 20771; 301-286-3713
james.mackinnon@nasa.gov

ABSTRACT

Due to their attractive benefits, which include affordability, comparatively low development costs, shorter development cycles, and availability of launch opportunities, SmallSats have secured a growing commercial and educational interest for space development. However, despite these advantages, SmallSats, and especially CubeSats, suffer from high failure rates and (with few exceptions to date) have had low impact in providing entirely novel, market-redefining capabilities. To enable these more complex science and defense opportunities in the future, small-spacecraft computing capabilities must be flexible, robust, and intelligent. To provide more intelligent computing, we propose employing machine intelligence on space development platforms, which can contribute to more efficient communications, improve spacecraft reliability, and assist in coordination and management of single or multiple spacecraft autonomously. Using TensorFlow, a popular, open-source, machine-learning framework developed by Google, modern SmallSat computers can run TensorFlow graphs (principal component of TensorFlow applications) with both TensorFlow and TensorFlow Lite. The research showcased in this paper provides a flight-demonstration example, using terrestrial-scene image products collected in flight by our STP-H5/CSP system, currently deployed on the International Space Station, of various Convolutional Neural Networks (CNNs) to identify and characterize newly captured images. This paper compares CNN architectures including MobileNetV1, MobileNetV2, Inception-ResNetV2, and NASNet Mobile.

I. INTRODUCTION

CubeSats (a subclass of SmallSats) were originally proposed as teaching tools and early technology demonstrations. However, since their inception with the space community, their role has matured, extending into more significant defense and science applications. This evolutionary trend towards more significant missions and goals led to a request from the National Aeronautics and Space Administration (NASA) and the National Science Foundation (NSF) to the National Academies of Sciences, Engineering, and Medicine to form a committee and conduct a review of the potential of the CubeSat platform and make key recommendations to improve the capabilities of the platform for future missions. The survey [1] published in 2016 concluded that CubeSats were already performing and meeting valuable science goals. However, while all space-science disciplines can benefit from CubeSat innovations, these small platforms cannot address or be a complete substitute for all platforms. The survey described that CubeSat systems "*excel at simple, focused, or short-duration missions and missions that need to be comparatively low cost or that require multi-point measurements.*"

The committee recommended focusing on maintaining low-cost approaches as the cornerstone of CubeSat development, while simultaneously stressing the importance and benefit of operating CubeSats and other SmallSats in swarms or constellations for multi-point measurements and extended spatial and temporal coverage. Combining these recommendations with the strict constraints (size, weight, power, and cost) of the small platform establishes a complex trade space to meet challenging science and defense goals.

In addition to the management and autonomy challenges of distributed satellite missions described in [1], many organizations also emphasize a distinct need for data analysis. The decadal strategy for Earth observation from space [2] highlights the need for studying large datasets captured by future constellations with semi-automated or autonomous capabilities for hazard detection and monitoring, hazard mapping, and hazard forecasting. Similarly, in his keynote address to the Small Satellite

Conference, Robert Cardillo, director of the National Geospatial-Intelligence Agency (NGA) noted that, with increasing data and imagery, new emphasis needs to be placed on smart and efficient analysis. He stated that NGA has moved from *"... staring at pictures and reporting, to programming algorithms and automation to drive production* [3]." Finally, these key objectives for future missions are repeated in the Long-Term Science and Technology Challenges [4] described by the Air Force Space Command (AFSPC). Critical defense focus areas for AFSPC include the study of autonomous, deep-learning, and highly adaptive systems, with additional interagency collaboration for small-satellite technology and big-data analysis.

To enable these large, distributed spacecraft missions, numerous technological advances are required. This paper proposes one such advance to benefit spacecraft autonomy and analysis capability, through the demonstration and planned flight verification of machine learning (ML) on CubeSat-scale processors. To achieve future mission objectives, more intelligent and capable computing can mitigate some of these challenges:

> "*Onboard data processing, autonomous systems, and navigation could further reduce the burden and cost of the ground segment and mission operations in CubeSats* [1]."

II. BACKGROUND

This section provides a cursory overview to Artificial Intelligence (AI) concepts and their uses in space computing. Additionally, frameworks and models for machine learning, a critical component for AI, are discussed. This section also describes the challenges for space computers to effectively run machine-learning applications. Finally, several AI-related designs and projects are highlighted.

Benefits for Space Applications

The concept of applying general, artificial-intelligence (AI) techniques is not novel and has been proposed for several decades. In [6], Girimonte from the European Space Agency (ESA) surveys several research areas of AI for space applications, specifically: distributed artificial intelligence (including swarms); large data analysis; enhanced situation self-awareness; and decisions support for spacecraft system design. AI is a broad subject, so for simplicity this paper highlights more recent examples of AI applications in three relevant categories: autonomy; communications; and analysis.

Spacecraft Autonomy is widely studied and includes a broad spectrum of topics such as navigation, coordination, planning and scheduling, and even reliability. Mission designers desire autonomy for a variety of reasons. One paramount motivator is round-trip communication delay time between an operator and a satellite. In scenarios, where the delay time of an operator responding is considerable, the spacecraft must be able to autonomously make decisions. Moreover, these intelligent systems can help improve spacecraft reliability by being trained to react to unexpected situations and guide the spacecraft to safer operational states with autonomous decision-making. Prominent examples are demonstrated by the Mars rovers. *Spirit*, one of the two rovers which landed on Mars in 2004, has software called *AutoNav* for terrain assessment to autonomously detect hazards based on imagery [7]. The *Opportunity* rover and the ChemCam spectrometer of *Curiosity* use automated data-collection software called AEGIS (Autonomous Exploration for Gathering Increased Science) to autonomously select high-value science targets [8]. Autonomy is also critically essential for future deep-space exploration, because these spacecraft may be outside communication range for extended periods of time and will encounter unknown environmental conditions, requiring the need to react accordingly. Chien describes flight software to enable onboard autonomy for deep-space exploration in [9]. Finally, intelligence can assist in coordinating and managing large swarms of spacecraft without causing the number of necessary ground operators to scale linearly as the constellation sizes increase. Coordination of swarms is described as the "fleet-management" problem in [1].

AI systems can be trained to reduce transmission bandwidth and processing on spacecraft by recognizing and capturing sensor data with pertinent information and discarding ineffectual ones. For spacecraft communications, such a requirement is essential to improve the efficacy of the (possibly erratic) communication link between a satellite and its ground station. There are two relevant examples of using machine learning to improve communication. The first is called MEXAR2 [10] (Mars Express AI Tool) and is used to determine the best schedule to optimize the timing of transmitted data packets to improve downlink capability. The second significant example is the Space Communications and Navigation (SCaN) Testbed [11] aboard the International Space Station (ISS). This experiment is designed to explore cognitive radio, which uses AI to find underused portions of the electromagnetic spectrum for communication.

As described previously, machine intelligence can also apply to performing on-board analysis for Earth-observation tasks. These tasks typically include hazard analysis (e.g. fire and flood detection), target detection, area monitoring, and weather forecasting. In [12],

researchers at NASA Goddard used ML to detect wildfires on MODIS (Moderate-resolution imaging spectroradiometer) data.

TensorFlow and TensorFlow Lite

There is an abundance of terrestrial research and development into employing AI for everyday life, such as self-driving automobiles. TensorFlow [13] is a popular, open-source, machine-learning framework developed by Google for research on many of the latest autonomous systems. In late 2017, Google released the developer preview of TensorFlow Lite, a framework for ML inference on embedded devices. The challenge for space vehicles also adopting such software frameworks is that these ground-based applications are typically executed on powerful CPU processors or GPU co-processors with high performance and maintainability. Small spacecraft, and CubeSats specifically, face challenges imposed by platform constraints on size, weight, and power, which limit processing capability, and prevent them from easily adapting the same designs.

Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a type of type of neural network most commonly used to analyze visual data. One of the very first CNNs, named as LeNet-5, was proposed in 1998 [14]. The core components of a CNN are small matrices of "weights." A convolutional layer in a CNN consists of one or more of these matrices. The output of one of these layers is the result of the convolution of the layer's kernels with the input. Typical CNNs use an activation layer after a convolutional layer. Activation layers apply a nonlinearity function to allow the network to approximate nonlinear functions. Some activation functions have been shown to decrease the time required to train some networks [15]. Popular activation functions include sigmoid, tanh, and the rectified linear unit (ReLU) [16]. The final layers of a typical CNN tend to be "fully connected" layers, which act as classifiers by reducing the feature map from the convolutional layers to a vector of output classes by a series of matrix multiplications. The output with the largest activation value is chosen as the result. The weights of each convolutional kernel as well as each fully connected layer are learned during the training process via backpropagation [17]. CNNs have emerged as the leader in image-processing tasks with Machine Learning since 2012 [18]. Extensive research has been performed to determine the optimal architecture for CNNs [19-21]. Figure 1 shows a basic CNN architecture.

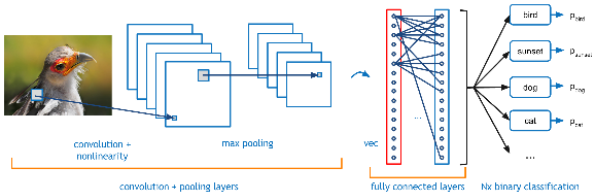

Figure 1: A Basic CNN Architecture*

In practice, the use of CNNs is composed of two main tasks: training and inference. Training is the process of "learning" the optimal set of weights that maximize accuracy of the desired task (e.g. image classification, object detection, semantic segmentation). Training is a highly compute-intensive process often accelerated by GPUs. Inference is the process of using a trained model (where parameters are no longer modified) to make decisions on novel data. Inference is a less compute-intensive process than training and has be performed on CPUs, GPUs, and FPGAs.

Computing Challenge

The most defining challenges for more advanced and capable artificial intelligence on satellites stem from the constraints imposed by small spacecraft computers. Unfortunately, due to the hazards of a radiation-filled space environment, radiation-hardened (rad-hard) computers are most commonly used in critical missions. However, these rad-hard computers are prohibitive due to cost and capability. Rad-hard devices are too expensive for missions, like CubeSats, that prioritize cost, and because they are expensive to develop, are typically outdated in both performance and features when compared to state-of-the-art commercial designs. Alternatively, mission developers can choose to fly commercial devices, which offer improved performance and energy efficiency over rad-hard devices but are susceptible to radiation effects. An overview of SmallSat computing and related challenges can be found in [22].

Consequently, these computing limitations are particularly challenging to ML because a significant amount of progress in deep learning and modern networks has been specifically conducted using GPUs. Many state-of-the-art network models require high-end GPU devices to run in inference, and even more capability to train. While there is some progress towards developing these networks for mobile applications (phones specifically), the most impressive results are attributed to high-end GPU systems [20]. Deep network models require significant amounts of processing

*https://adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

capability for matrix operations, and extensively strain the memory bandwidth and capacity of even the most capable systems.

As described in [22], modern space computers would struggle to meet the minimum requirements for complex, deep-learning architectures. Additionally, there are a scarce number of GPUs that have been evaluated to work in a space environment, while simultaneously meeting the low-power restrictions of SmallSat platforms. These computing challenges are further emphasized by Robert Laudati, the managing director of commercial products at Harris Space and Intelligence Systems. He comments that the future is to move more computing to space (with onboard computing), and that "*he does not see that capability coming to the market any time soon.*"†

Related Research

Despite the considerable challenge posed by the computational requirements of ML, there are several related works that explore the state-of-the-art networks for embedded systems and satellites. In [23], Schartel trained the SqueezeNet model on a terrestrial system and planned to transfer the model to an embedded system; however, the entire design was not fully implemented. In [24], researchers at the University of New Mexico partnered with Stinger Ghaffarian Technologies and Air Force Research Laboratory Space Vehicles Directorate to demonstrate image classification on the Nvidia TX1. In their demonstration, a desktop GPU is used to train the model, and inference is performed on the TX1 with the CUDA Deep Neural Network (cuDNN) library and TensorRT. Lastly, in [25], SRC Inc., developed their own deep CNN framework for use on a Xilinx Artix-7 FPGA platform. With their design, they studied image classification and compared their results against the IBM TrueNorth NS1e development board, a neuromorphic computer with machine-learning capabilities.

III. APPROACH

In comparison to related research, our approach focuses on developing a machine-learning solution that can run on existing flight hardware with TensorFlow. For our testbed and experiment, we focus on the Xilinx Zynq-7020 which is the featured technology of the CSPv1 flight computer described in [22]. To test the computational capability of the Xilinx Zynq-7020 for ML inference, we trained CNNs for image classification and benchmarked the accuracy, execution time, and runtime memory usage of four target CNN architectures on the Digilent ZedBoard development system.

Dataset

Our dataset consists of images collected by our flight system on the ISS. Our mission, known as STP-H5/CSP [26] launched on the SpaceX CRS-11 in February 2017. Since its launch, STP-H5/CSP has been collecting and downlinking images. Over the past year, we have downloaded approximately eight thousand thumbnails, each a 489×410 pixel image. The images from CSP were used to create a small dataset to train image-classification models. Most of the images depict one of five classes: black (Example of Images in Each ClassFigure 2a); cloud/water (Figure 2b); distorted (Figure 2c); land (Figure 2d); or white (Figure 2e). Each of the 8000 images was downloaded from CSP and labeled as one of the classes cited above.

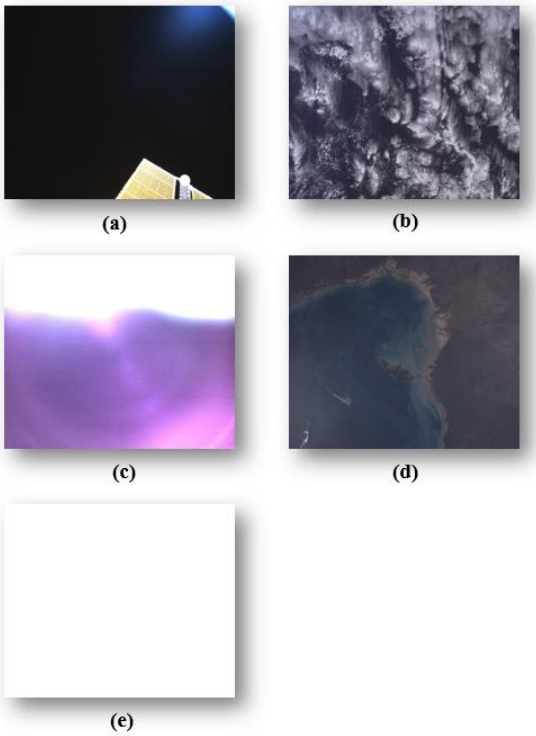

Figure 2: Example of Images in Each Class

Transfer learning is the process of using a trained ML model to bootstrap a model for a related task. In the case of a CNN, transfer learning means freezing previously trained weights for convolution layers and only learning the weights for the classification layers [27]. Despite having thousands of images in the STP-H5/CSP collection, this data is considered limited for training deep CNNs. Thus, training a CNN such as MobileNet or Inception from scratch with only this limited dataset was

† http://spacenews.com/artificial-intelligence-arms-race-accelerating-in-space/

deemed impractical. However, transfer learning provides a method to use a relatively small dataset in the training process. To bootstrap our models, we used CNNs pre-trained on ImageNet, a massive, industry-standard dataset for image classification [19].

Target CNN Architectures

We compare the classification accuracies of four modern CNN architectures (MobileNetV1, MobileNetV2, Inception-ResNetV2, and NASNet Mobile) on our dataset. Table 1 shows the reported top-1 and top-5 accuracies of each target architecture and their variants on ImageNet data.

Table 1: ImageNet Image Classification Accuracies of Relevant CNN Architectures

Network	Top-1 Accuracy	Top-5 Accuracy
MobileNetV1	70.6%	89.5%
MobileNetV2	74.7%	92.5%
GoogLeNet	-	93.3%
ResNet	80.6%	96.4%
Inception-ResNetV2	80.1%	95.1%
NASNet	82.7%	96.2%
NASNet Mobile	74.0%	91.6%

MobileNetV1 was developed by Google in 2017. It is considered a "mobile-first" (emphasizing phones and embedded devices primarily) CNN architecture, designed to be more efficient for inference than a typical CNN. It replaces standard convolutions with depthwise-separable convolutions. This approach drastically reduces the number of trained parameters, which reduces model size and improves inference performance [28]. MobileNetV2 [29] is a revision of MobileNetV1 which adds inverted residuals and linear bottleneck connections. Both versions of MobileNet use two hyperparameters, a width multiplier and a resolution multiplier, to specialize the architecture. The width multiplier is a scaling factor applied to the number of convolution filters in each layer of the network. The typical values for the width multiplier are 0.25, 0.50, 0.75, and 1.0 for MobileNetV1. The resolution multiplier is a scaling factor applied to the size of the input image to the network. The typical values for the input image resolution are 128, 160, 192, and 224.

Inception-ResNetV2 was also developed by Google and combines the architectures of GoogLeNet, the winner of the ILSVRC (ImageNet Large Scale Visual Recognition Challenge) in 2014, and Microsoft's ResNet, the 2015 ILSVRC winner [19]. Inception-ResNetV2 is a GoogLeNet architecture with Inception Modules and residual connections. Inception Modules use a combination of 1×1, 3×3, and 5×5 convolutions as well as 3×3 max pooling with dimension reductions via 1×1 convolutions to lower computational complexity [19]. Residual connections allow layers to fit a residual identity mapping between layers [19].

NASNet is a product of Google's AutoML project. NASNet is inspired by the Neural Architecture Search (NAS) framework which uses a reinforcement learning search method to optimize architecture configurations [30]. The largest NASNet variant achieved the highest published accuracy to date on ImageNet image classification [30]. NASNet Mobile is a smaller variant of NASNet.

As a starting point for re-training the target CNNs, we used Google's TensorFlow Hub models[‡]. TensorFlow Hub is a collection of pre-trained models that can be used for transfer learning and was released by Google in 2018.

Our dataset of 8000 images was divided into three sets, training (70%), validation (10%), and testing (20%). Each network was trained for 500 epochs with a learning rate of 0.01 and a batch size of 100 images.

IV. RESULTS

In this section we present results of our studied networks on our image dataset. We compare the results based on accuracy of the network, followed by performance, which is essential to embedded space systems.

Accuracy Results

For our displayed results, we measured the top-1 (prediction from the model matches the image label) and top-2 (either of the two highest-probability predictions from the model match the image label) accuracies of each transfer-learned CNN on the test set. Each MobileNetV1 and MobileNetV2 variant (all values for width and resolution multipliers) was trained, however, for brevity we only present the most accurate variants.

[‡] https://www.tensorflow.org/hub/

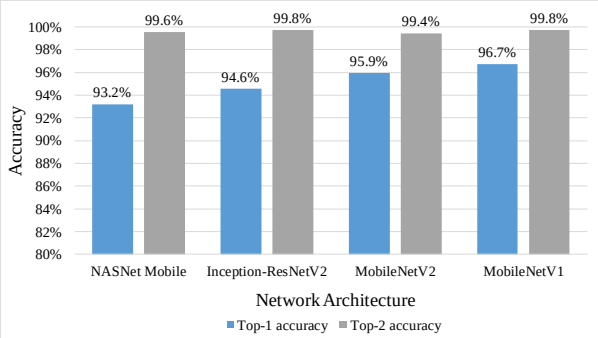

Figure 3: CNN Accuracy on STP-H5/CSP Images

Each CNN performed adequately on the dataset, achieving over 90% top-1 accuracy and near-perfect top-2 accuracy, as shown in Figure 3. MobileNetV1 outperformed the other CNNs, despite having the worst accuracy on ImageNet in the set. It is worth noting that the most-accurate MobileNetV1 variant was with the width multiplier 1.0 and input image resolution 224×224 (i.e. no reduction in the number of convolution filters or input image resolution).

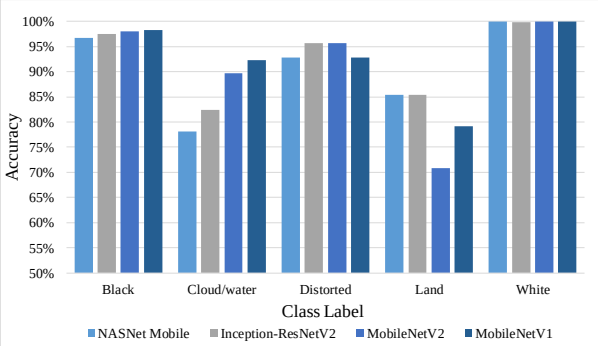

Figure 4: Per-class Accuracy on STP-H5/CSP Images

In addition to top-1 and top-2 accuracy, we measured how accurate each model was on each of the classes in our STP-H5/CSP dataset as displayed in Figure 4. For the black, distorted, and white classes, each model performed well as these classes are distinctive, with little overlap in features. The cloud/water and land classes, however, are more difficult for classification. There is similarity between the cloud/water and land classes, making it difficult for all tested models to distinguish between the classes consistently, specifically because many images contain some land, water, and clouds. MobileNetV1 is the only architecture that achieved over 90% accuracy on cloud/water images; it additionally maintained nearly 80% accuracy on land images. NASNet Mobile and Inception-ResNetV2 performed best on land images, but both struggled with cloud/water images. Finally, MobileNetV2 performed well on cloud/water images, at the expense of low land-image accuracy.

Performance Results

For our on-board performance analysis, we focused on MobileNetV1 because it was the most accurate CNN on the STP-H5/CSP test dataset. Using TensorFlow Lite, we performed inference on all MobileNetV1 variants. We also measured the execution time required to classify an image and the amount of memory used during classification. All tests were conducted on the Digilent ZedBoard, which is regularly used as a facsimile development kit for the CSPv1 flight computer.

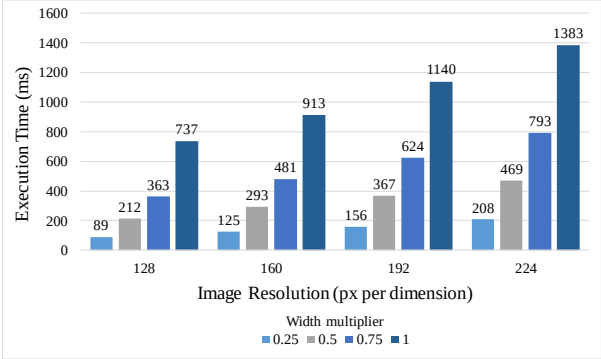

Figure 5: MobileNetV1 Execution Time on ZedBoard

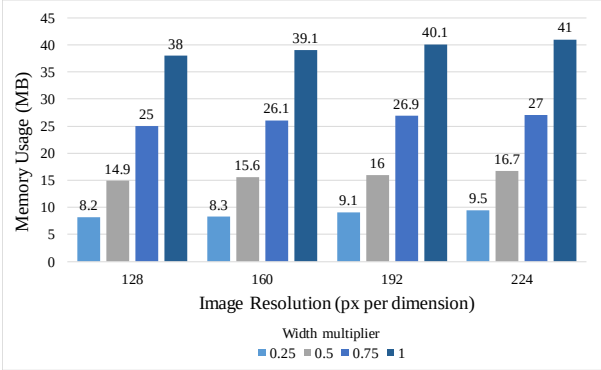

Figure 6: MobileNetV1 Memory Usage on ZedBoard

Both execution time (Figure 5) and memory usage (Figure 6) scale linearly with respect to the number of pixels in the input image and quadratically with the width multiplier. The width multiplier (i.e. the number of convolution filters in each layer) has a larger effect than image resolution on both execution time and runtime memory usage. The smallest MobileNetV1 variant (width multiplier 0.25 and input image resolution 128x128) achieves 11 FPS on the Zynq-7020 while using just 8 MB of RAM. Our performance satisfies mission

requirements and falls well within the memory constraints of our space system.

V. CONCLUSIONS

SmallSats in general and CubeSats in particular face arduous challenges in achieving more significant science and defense goals. To meet new mission objectives, on-board data analysis is rapidly becoming the key focus area for SmallSat development. AI systems can enable more efficient use of some system resources and perform crucial processing tasks for autonomous operation on a spacecraft. However, modern ML frameworks are typically executed on resource-intensive GPUs, making their deployment on these space systems very limited.

Using a dataset of collected space images from our STP-H5/CSP mission on the ISS, this paper demonstrates that we can achieve reasonable performance with modern ML models on a low-memory, low-power, space-grade, embedded platform. Our results show it would be feasible for the TensorFlow Lite framework to be used for deploying deep-learning models in future space missions on similar space-computing platforms. Additionally, leveraging CNNs pre-trained on ImageNet is shown to be effective for image-classification tasks on terrestrial-scene images.

Future Work

This research establishes the foundation towards additional extensions into AI-capable small spacecraft. The immediate next step is to upload the inferred CNNs directly onto the STP-H5/CSP system, thereby enabling us to filter undesirable images (i.e. images classified as white, black, and distorted) in real-time. Thus, AI can prevent the system from wasting bandwidth by sending insignificant images. To extend the classification, more complex image-processing tasks will be studied, such as object detection and semantic segmentation. Since our NSF SHREC Center is regularly proposing new missions and apps, this research can be used for more complex science classifications with smaller GSD (Ground Sample Distance) technologies to be featured on future mission proposals. Finally, future extensions could include adding accelerated TensorFlow Lite inference operations using FPGAs (e.g., in CSP) and incorporating other hardware accelerators within the design.

Acknowledgments

This research was funded by industry and government members of the NSF SHREC Center and the National Science Foundation (NSF) through its IUCRC Program under Grant No. CNS-1738783. The authors would also like to thank additional CSP team contributors including Sebastian Sabogal and Antony Gillette, and NASA Goddard contributors Troy Ames and Dan Mandl.

References

1. Board, S. S., and National Academies of Sciences, Engineering, and Medicine, *Achieving Science with CubeSats: Thinking Inside the Box*, Washington, DC: National Academies Press, 2016.
2. National Academies of Sciences, Engineering, and Medicine, *Thriving on Our Changing Planet: A Decadal Strategy for Earth Observation from Space*. Washington: National Academies Press, Jan 2018.
3. Cardillo, R., "Small Satellite 2017 Keynote Address," 31st Annual AIAA/USU Conference on Small Satellites, Logan, UT, Aug 7, 2017. https://www.nga.mil/MediaRoom/SpeechesRemarks/Pages/Small-Satellites---Big-Data.aspx
4. Sanchez, M., "AFSPC Long-Term Science and Technology Challenges," Space and Cyberspace Innovation Summit, Aug 23-24, 2016. http://www.defenseinnovationmarketplace.mil/resources/Innovation_Summit_Phase1_Intro.pdf
5. Copeland, M., "What's the Difference Between Artificial Intelligence, Machine Learning, and Deep Learning?," NVIDIA Blog, Jul 29, 2016. https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/
6. Girimonte, D and D. Izzo, "Artificial Intelligence for Space Applications," in *Intelligent Computing Everywhere*, Springer, London, 2007, pp. 235–253
7. Leger, C., Trebi-Ollenu, A., Wright, J., Maxwell, S., Bonitz, R., Biesiadecki, J., Hartman, F., Cooper, B., Baumgartner, E., and M. Maimone, "Mars exploration rover surface operations: Driving Spirit at Gusev crater," 2005 IEEE Conference on Systems, Man, and Cybernetics, Oct 2005, pp. 1815-1822.
8. Estlin, T., Bornstein, B., Gaines, D., Anderson, R. C., Thompson, D., Burl, M., Castano, R., and M. Judd, "AEGIS automated targeting for the mer opportunity," ACM Transactions on Intelligent Systems and Technology, Vol. 3, No. 3, May 2012, pp. 1–19.
9. Chien, S., Bue, B., Castillo-Rogez, J., Gharibian, D., Knight, R., Schaffer, S., Thompson, D. R., and K. L. Wagstaff, "Agile science: Using onboard autonomy for primitive bodies and deep space exploration," Proceedings of the International Symposium on Artificial Intelligence, Robotics, and Automation for Space, Montreal, Canada, Jun 2014.
10. Cesta, A., Cortellessa, G., Denis, M., Donati, A., Fratini, S., Oddi, A., Policella, N., Rabenau, E., and

J. Schulster, "Mexar2: AI Solves Mission Planner Problems," IEEE Intelligent Systems, Vol. 22, No. 4, July 2007, pp. 12–19.

11. Reinhart, R., "Space Communications and Navigation Testbed: Communications Technology for Exploration," ISS Research and Development Conference, Denver, CO, Jul 16-18, 2013.
12. Mackinnon, J., Ames, T., Mandl, D., Ichoku, C., Ellison, L., Manning, J., and B. Sosis, "Classification of Wildfires from MODIS Data using Neural Networks," Machine Learning Workshop, Aug 2017.
13. Abadi, M., Barham, P, Chen, J., Chen, Z, Davis, A., Dean, J., Devin, M., Ghemawat, S., Iving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and X. Zheng, "TensorFlow: A System for Large-Scale Machine Learning," 12th USENIX Syposium on Operating Systems Design and Implementations, Savannah, GA, Nov 2-4, 2016.
14. Lecun, Y., Bottou, L., Bengio, Y., and P. Haffner, "Gradient-based learning applied to document recognition," Proceedings of the IEEE, Vol. 86, No. 11, Nov 1998, pp. 2278-2324.
15. Clevert, D. A., Unterthiner, T., and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," arXiv preprint arXiv:1511.07289, 2015.
16. Nair, V., and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," Proceedings of the 27th international conference on machine learning (ICML-10), Haifa, Israel, Jun 21-24, 2010, pp. 807-814.
17. Rumelhart, D. E., Hinton, G. E., and R. J. Williams, "Learning representations by back-propagating errors," Nature International Journal of Science, Vol. 323, No. 6088, Oct 1986, pp. 533–536.
18. Krizhevsky, A., Sutskever, I., and G. Hinton, "ImageNet classification with deep convolutional neural networks," Proceedings of the 25th International Conference on Neural Information Processing Systems, Vol. 1, Dec 2012, pp. 1097-1105.
19. He, K., Zhang, X., Ren, S., and J. Sun, "Deep residual learning for image recognition," arXiv:1512.03385, 2015.
20. Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size," arXiv preprint arXiv:1602.07360, 2016.
21. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and A. Rabinovich, "Going Deeper with Convolutions," arXiv:1704.04861, 2014.
22. George, A. D., and C. Wilson, "Onboard Processing with Hybrid and Reconfigurable Computing on Small Satellites," Proceedings of the IEEE, Vol. 106, No. 3, Mar 2018, pp. 458–470.
23. Schartel, A., "Increasing Spacecraft Autonomy through Embedded Neural Networks for Semantic Image Analysis," M.S. thesis, Luleå University of Technology, Luleå, Sweden, 2017.
24. Buonaiuto, N., Kief, C., Louie, M., Aarestad, J., Zufelt, B., Mital, R., Mateik, D., Sivilli, R., and A. Bhopale, "Satellite Identification Imaging for Small Satellites Using NVIDIA," Proceedings of the AIAA/USU Conference on Small Satellites, Launch, SSC17-WK-56.
25. Grabowski, J., "Neuromorphic Computing for Classification in Optical and SAR Data with IBM TrueNorth and Xilinx FPGAs," 10th Workshop on Fault-Tolerant Spaceborne Computing Employing New Technologies, Albuquerque, NM, May 30 – Jun 2, 2017.
26. Wilson, C., Stewart, J., Gauvin, P., MacKinnon, J., Coole, J., Urriste, J., George, A., Crum, G., Wilson, A., Wirthlin, M., "CSP Hybrid Space Computing for STP-H5/ISEM on ISS," 29th Annual AIAA/USU Conf. on Small Satellites, Logan, UT, August 8-13, 2015.
27. Pan, S., and Q. Yang, "A Survey on Transfer Learning," IEEE Transactions on Knowledge and Data Engineering, Vol. 22, No. 10, Oct 2010, pp. 1345–1359.
28. Howard, A., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and H. Adam. "Mobilenets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.
29. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., and L.-C. Chen. "MobileNetV2: Inverted residuals and linear bottlenecks," arXiv preprint arXiv:1801.04381, 2018.
30. Zoph, B., Vasudevan, V., Shlens, J., and Q. V. Le., "Learning transferable architectures for scalable image recognition," arXiv preprint arXiv:1707. 07012, 2017.