Kodan: Addressing the Computational Bottleneck in Space

Bradley Denby
bdenby@andrew.cmu.edu
Carnegie Mellon University, USA

Krishna Chintalapudi
krchinta@microsoft.com
Microsoft Research, USA

Ranveer Chandra
ranveer@microsoft.com
Microsoft Research, USA

Brandon Lucia
blucia@andrew.cmu.edu
Carnegie Mellon University, USA

Shadi Noghabi
shadi.noghabi@microsoft.com
Microsoft Research, USA

ABSTRACT

Decreasing costs of deploying space vehicles to low-Earth orbit have fostered an emergence of large constellations of satellites. However, high satellite velocities, large image data quantities, and brief ground station contacts create a data downlink challenge. Orbital edge computing (OEC), which filters data at the space edge, addresses this downlink bottleneck but shifts the challenge to the inelastic computational capabilities onboard satellites. In this work, we present Kodan: an OEC system that maximizes the utility of saturated satellite downlinks while mitigating the computational bottleneck. Kodan consists of two phases. A one-time transformation step uses a reference implementation of a satellite data analysis application, along with a representative dataset, to produce specialized ML models targeted for deployment to the space edge. After deployment to a target satellite, a runtime system dynamically selects the best specialized models for each data sample to maximize valuable data downlinked within the constraints of the computational bottleneck. By intelligently filtering low-value data and prioritizing high-value data for transmit via the saturated downlink, Kodan increases the data value density between 89 and 97 percent.

CCS CONCEPTS

• **Hardware** → **Analysis and design of emerging devices and systems**; • **Computer systems organization** → **Embedded systems**; • **Software and its engineering** → *Software performance*; • **Networks** → *Network performance evaluation.*

KEYWORDS

embedded systems, IoT, edge computing

ACM Reference Format:
Bradley Denby, Krishna Chintalapudi, Ranveer Chandra, Brandon Lucia, and Shadi Noghabi. 2023. Kodan: Addressing the Computational Bottleneck in Space. In *Proceedings of the 28th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 3 (ASPLOS '23), March 25–29, 2023, Vancouver, BC, Canada.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3582016.3582043

ASPLOS '23, March 25–29, 2023, Vancouver, BC, Canada

ACM ISBN 978-1-4503-9918-0/23/03.
https://doi.org/10.1145/3582016.3582043

1 INTRODUCTION

The proliferation of commercial space launch services [15] and nanosatellites [32, 36] over the past two decades makes low-Earth orbit (LEO) accessible to deployments of state-of-the-art, sensor-equipped computer systems inside satellites. These satellites enable new and valuable geospatial sensing and computing applications, including disaster relief [18], agriculture [40], and infrastructure monitoring. The Earth-observation market remains dominated by large-scale, monolithic satellites costing hundreds of millions of US dollars each. However, the ascendance of inexpensive nanosatellites has led to large, commercial *constellations* of nanosatellites in LEO [5, 26]. Lower device cost and higher launch cadence decreases risk and enables new satellite applications [30].

Communication and computation ability constrain satellite utility. Today, most satellites operate as "bent pipes" [25] and are tasked manually to collect and downlink raw sensor data. These satellites face a *downlink bottleneck* stemming from a lack of communication opportunities. Recent work on "orbital edge computing" (OEC) processes data on satellites before downlinking [7, 8]. OEC mitigates the downlink bottleneck by identifying signals of interest and downlinking those signals only. While OEC addresses the downlink bottleneck, we observe that edge processing creates a *computational bottleneck* limiting the value of each satellite.

Processing satellite sensor data at the edge is challenging. Satellite sensor samples (e.g., images) are large and arrive at a high rate compared to the rate at which embedded satellite hardware can process them. Geospatial image frames can contain hundreds of square kilometers and hundreds of millions of pixels. Depending on orbit altitude and camera characteristics, a LEO, Earth-observation satellite observes an entirely new frame every $1-30$ seconds (the "frame deadline"). Applications typically *tile* these large frames and process each tile on the ground using, e.g., machine learning algorithms. However, not all samples are equally valuable; some observations are of *high-value* to an application and others are of *low-value*. A system faced with a saturated link should prioritize transmission of high-value data, but today's bent pipes send data indiscriminately. Processing samples at the space edge distinguishes these categories. The computational bottleneck stems from an inability to process all tiles within the frame deadline. Prior OEC work manages the computational bottleneck by statically distributing tile processing across a constellation, using satellite-parallelism to meet the frame deadline [8]. Such a scheme distributes work across hundreds of satellites while addressing the computational bottleneck for just a single application — a relatively high-cost solution resulting in a vertically-integrated constellation aimed at a particular purpose.

Kodan is an orbital edge computing system that maximizes data value from satellites limited by communication and computation. Under a saturated satellite downlink, Kodan mitigates the computational bottleneck without the high cost of hundreds of satellite-parallel processors. Kodan uses a combination of techniques that modify applications based on unique, orbital data characteristics by trading between geospatial analysis precision and processing speed. After deployment to a satellite, the Kodan runtime system dynamically selects appropriate optimizations for each observation. Kodan decides how to process a sample based on its *geospatial context*. A geospatial context is a property of a data sample indicating its likelihood to contain certain features, e.g., the presence of ocean, forest, tundra, clouds, or high-value data. High-precision value labels are computationally easier in some contexts and harder in others.

Kodan balances precision and execution time to maximize data value density. Software running on each satellite prioritizes decreased compute time when computationally bottlenecked and prioritizes precision otherwise. When computationally limited, Kodan uses tile context to select an action: the satellite downlinks data in high-value contexts, discards data in low-value ones, and executes an application to more thoroughly filter the rest. Whether or not computationally limited, Kodan uses context-specific models to increase precision and downlinked *data value density*, i.e., the fraction of a saturated downlink composed of high-value bits. Kodan recognizes that not all sensor data need equal care in processing. To trade precision for execution time, Kodan adjusts frame tile count to reduce data quantity at a cost in quality.

Kodan increases downlinked data value density by mitigating the computational bottleneck. To show the value of Kodan, we implement seven end-to-end, deployment-ready, pixel-segmentation applications trained to filter low-value clouds using publicly-available, geospatial datasets. We quantify the improvement in valuable data downlinked with on-orbit computing using context-specialization to identify high-value observations; Kodan improves the data value density of the saturated downlink between 89 and 97 percent compared to the bent pipe.

To summarize, the main contributions of this work are:

- We characterize the impact of orbital edge computing on both the downlink bottleneck and the computational bottleneck.
- We present Kodan, an OEC system that addresses the computational bottleneck using hardware-aware modifications of satellite applications.
- We characterize and evaluate context-specific models, frame tiling, and context-based elision to maximize data value density within the constraints of the computational bottleneck.
- We provide a comprehensive evaluation of Kodan across satellite data processing applications and hardware targets, resulting in improvements to the data value density of the saturated downlink between 89 and 97 percent.

2 BACKGROUND AND MOTIVATION

We provide context for Kodan and characterize challenges for Earth-observation satellites.

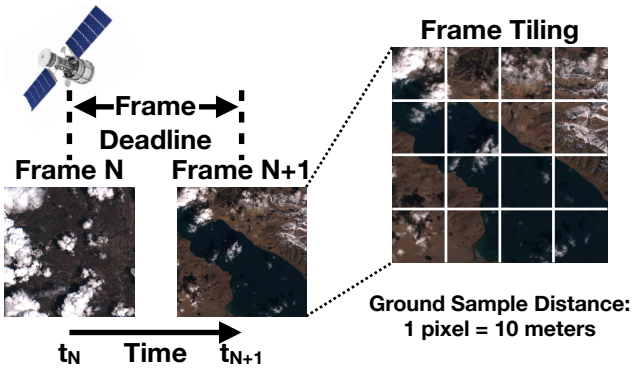

Figure 1: A satellite periodically captures image frames, and the time between frame captures is the frame deadline. Before processing an image, geospatial analysis software splits a frame into tiles. Details visible in the images depend on the ground sample distance, or the geographic area per pixel.

Earth-observation satellites: LEO, Earth-observation satellites collect sensor data — e.g., multispectral images — for geospatial analytics. These satellites often deploy to polar orbits (i.e., orbits crossing near the poles of Earth). As the satellite travels through its orbit, it accesses nearly all latitudes; as the planet rotates, the satellite accesses all longitudes. LEO altitudes are hundreds of kilometers, and LEO periods are about 90 min.

Historically, Earth-observation satellites are large and monolithic. Recently, inexpensive nanosatellites have proliferated. The monolithic Worldview [13], Earth-Observing 1 (EO-1) [33], and Landsat [27] satellites cost hundreds of millions of US dollars each (e.g., $855,000,000 [20]). Now, many missions use cubesats [32], chipsats [42], and pocketqubes [9, 36] to increase hardware refresh cadence and avoid the high costs of monolithic satellites. Lower costs enable Earth-observing *constellations* consisting of hundreds of devices [5, 26].

Orbital mechanics determine both access to and the quality of satellite sensor data. For images, a satellite captures a *frame* along its *ground track*. A frame is a large geographic region; geospatial applications often split frames into many smaller *tiles* for analysis. Satellite image quality is characterized by *ground sample distance* (GSD) — geographic distance between adjacent pixels — which may range from km/px to cm/px [13, 27] and is determined by altitude and camera characteristics. Figure 1 illustrates these concepts.

The bent pipe: Today, most Earth-observation satellite operators manually task their devices to sense and downlink raw observations to a datacenter for processing, i.e., a *bent pipe* [12, 25]. Communication opportunities for high-velocity, LEO satellites last only for a few minutes while the device is near a ground station and may occur infrequently depending on the orbit. State-of-the-art communication systems downlink a total data quantity of MBs or GBs per pass. This *downlink bottleneck* constrains observation rate because not all data can be sent (we quantify this bottleneck in Section 2.1).

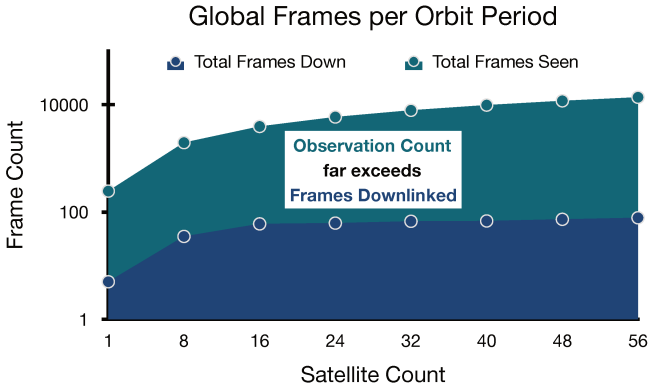

Figure 2: A single satellite observes more frames per revolution than it downlinks. As satellite count increases, contention and eventually ground segment saturation widens this gap. Note the y-axis log scale.

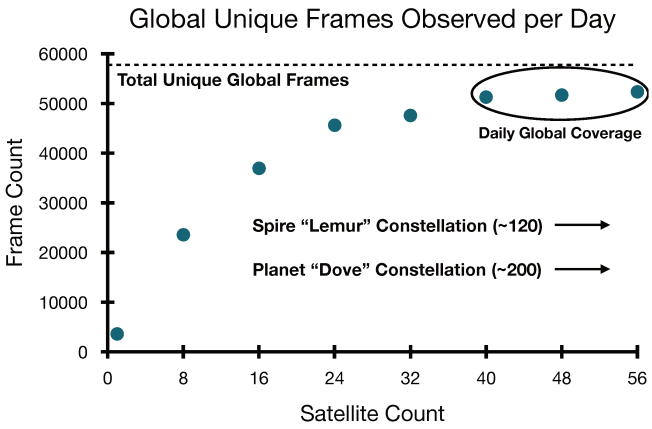

Figure 3: For the first time, constellations exist with enough satellites to achieve *daily global coverage*. Saturated downlinks lock these daily global observations at the orbital edge.

Orbital edge computing: Recent work proposes orbital edge computing (OEC) [7, 8], in which satellites process data at the space edge. OEC distributes computation across a constellation where each satellite contains highly-capable, commercial, off-the-shelf (COTS) compute hardware rather than low-performance, radiation-hardened, space CPUs. An OEC satellite tiles each image frame and processes tiles to identify interesting data to transmit. Especially when interesting features are rare, OEC addresses the downlink bottleneck with edge computing by triaging sensor data before transmission. However, OEC must process all frame tiles before a new frame enters the sensor view, creating a *frame processing deadline* usually between $1 - 30$ s. Failing to meet the deadline is a *computational bottleneck*.

2.1 Challenges at the Orbital Edge

We highlight two orbital edge challenges: a downlink bottleneck prevents sending all raw data, and a computational bottleneck prevents processing all data on orbit. We quantify these bottlenecks using the `cote` [8] simulator to model existing satellites, including orbital dynamics, sensing, communication, and the ground segment. We validate our analysis with publicly-available satellite and ground segment performance metrics [27]. In this section, we address three key questions:

1. What limitations arise from today's downlink bottleneck, and what will its impact be in the future (Section 2.1.1)?
2. To what extent could edge computing address the downlink bottleneck (Section 2.1.2)?
3. How much of this potential improvement can be realized by directly deploying data processing applications to the space edge, and to what extent does the computational bottleneck limit these improvements (Section 2.1.3)?

2.1.1 The Downlink Bottleneck. Downlink capacity cannot support existing satellite sensor datarates, and the gap grows with sensor fidelity and constellation population. We quantify this gap for Landsat 8 in Figure 2. Over one orbit revolution, the ground segment supports reception of just 2% of the available observations of hyperspectral, 10K image frames. When satellite count in the same orbit plane increases from one to 16, downlinked data increases from 5 frames during one period to 60 frames during the same period; this improvement stems from claiming previously *idle ground station* time. When the Landsat ground segment serves one satellite, stations sit idle while the satellite is out of range (i.e., most of the time). Additional satellites, when not contending with each other, claim idle time and increase total downlinked data. However, as constellation population increases, the space segment eventually saturates the downlink. Adding satellites beyond this population count increases the ability to *observe* but not downlink additional data, widening the downlink gap.

Why are large constellations desirable? Larger constellations increase sensor coverage of Earth. Figure 3 shows the satellite count required for daily global coverage, i.e. the opportunity to observe all Landsat frames each day (see Section 2.1.1 for discussion on whether such observations could be downlinked). We add support to `cote` for the Landsat Path/Row World Reference System (WRS) [38] and import the WRS-2 scene boundary shapefiles to produce this plot. Reaching global daily coverage requires a constellation population of at least 40. Constellations in different orbits or with different sensors, like the Spire "Lemur" or Planet "Dove" satellites, deploy even greater numbers of devices.

2.1.2 Addressing the Saturated Downlink. Bent-pipe satellites waste limited downlink capacity by indiscriminately sending observations containing both high-value and low-value data. To demonstrate this fact, we examine a cloud-filtering application. On average, 67% of satellite images are obscured by clouds [23] and are low-value to most customers. For Landsat 8, Figure 4 (left column) shows that, during one day, just 1/3 of the data from nearly 3600 observable frames is high-value (i.e., not cloudy). Accounting for the downlink bottleneck (middle), less than 21% of observable high-value data is downlinked with a bent-pipe. Ideal edge filtering (100% accuracy and zero execution time) delivers over 3× more high-value data

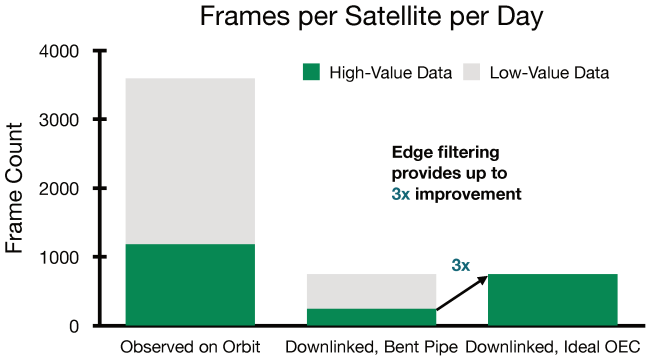

Figure 4: A single satellite observes more frames per day than can be downlinked per day. Many of these frames contain low-value data. Without OEC, a relatively small amount of downlinked data is high-value (21%). Identifying high-value data on orbit can improve the data value density by 3×.

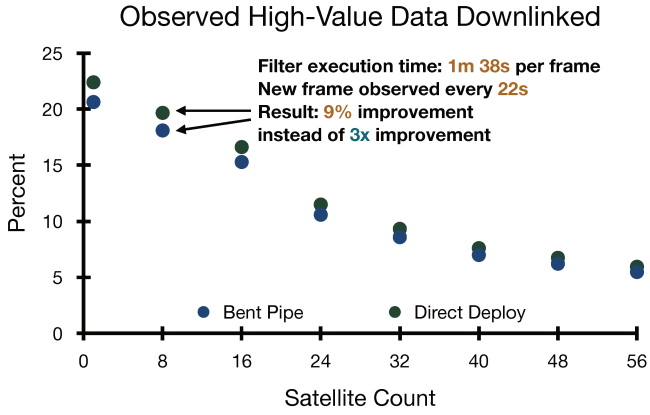

Figure 5: Directly deploying existing geospatial analysis applications to the inelastic orbital edge provides limited improvement in saturated downlink utilization. These applications are designed for elastic datacenters and cannot process data at the rate of new satellite observations.

— 63% of the total, observable high-value data, and the maximum possible under the downlink bottleneck (right column). Increasing the ratio of high-value to low-value data increases the *data value density* of the saturated downlink: the fraction of downlinked data composed of high-value bits.

2.1.3 The Computational Bottleneck. While ideal cloud filtering offers a potential 3× improvement in valuable data downlinked, the space edge must contend with the *inelastic and limited* computational resources on satellites. Volume, mass, energy, and cost constraints at the space edge prevent deployment of unlimited computational resources inside a satellite [8]. Unless a filtering application completes within the frame deadline, a satellite cannot process all frames, creating a *computational bottleneck* that limits the ability of OEC to address the downlink bottleneck. We quantify this effect for a real cloud filter application [31]. Figure 5 shows the fraction of high-value data downlinked with and without edge filtering for a range of constellation sizes. Direct-deployment of cloud filtering improves high-value data downlinked by just 9%, far short of the potential 3×. The shortfall stems from the 98 s frame processing time, which exceeds the 22 s frame deadline; only a fraction of captured frames can be filtered. Therefore, while OEC has potential to mitigate the downlink bottleneck, the computational bottleneck limits its benefit.

Limitations of parallel, distributed computation: Prior OEC work addresses computational needs by distributing work across a *pipeline* of satellites. While effective at reducing per-satellite compute time to meet full ground track coverage, pipeline populations must be very large (e.g., > 100 devices per application). This approach is costly and designed for vertically-integrated constellations deployed for a single purpose. In the future, we expect the emergence of constellations acting as a platform for customer applications, i.e., a constellation-as-a-service. As a result, the naturally inelastic space edge has pressure to operate at its computational limit in order to maximize platform value. Prior OEC work provides no technique to reduce per-satellite computational load without increasing constellation population; this shortcoming is a key motivation for Kodan.

3 KODAN SYSTEM DESIGN

Kodan is an OEC system maximizing the data value density of a saturated satellite downlink by adjusting geospatial analysis software to adapt to target satellite computing hardware in space. As constellation population increases to eliminate idle ground station time, the ground segment becomes *saturated*, and additional satellites increase observation opportunity but not downlinked data quantity. To increase the utility of this saturated downlink, the fraction of downlinked data composed of high-value bits — the *data value density* — must increase. Distinguishing between high-value and low-value sensor data on a satellite requires execution of geospatial analysis software at the space edge. Unless a satellite processes each sensor sample before the next data capture, not all observations are filtered: a *computational bottleneck*. Satellites are edge devices in space; volume, mass, energy, and cost constraints prevent deployment of unlimited computational resources [8]. Kodan reduces the computational requirements of geospatial analysis software deployed to the space edge.

Kodan makes different adjustments to an application's computational requirements for each hardware deployment target. This approach is an alternative to simply increasing the computational capability of a satellite — something that cannot be done for satellites already in space, and an option constrained by volume, mass, energy, and cost limitations. This approach is also an alternative to simply increasing the number of satellites in a constellation and distributing processing, which incurs high monetary cost and only addresses constellations executing a single application.

To adjust the computational needs of a geospatial analysis application, Kodan leverages three techniques: *context-specific models*, *frame tiling*, and *context-based elision*.

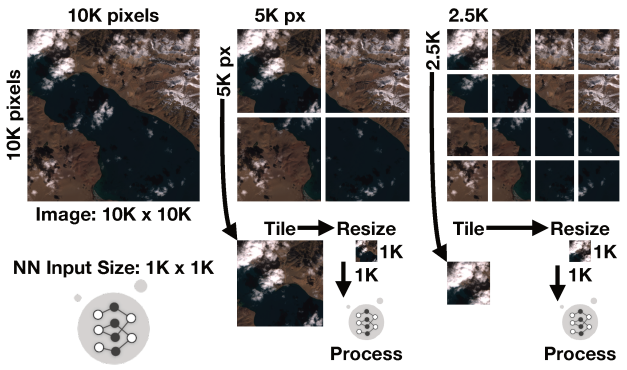

Figure 6: Tile count per frame determines frame processing time. A smaller tile count decreases frame processing time, but each tile must undergo more decimation to match the neural network input size. A larger tile count increases frame processing time, and each tile undergoes less decimation.

Context-based model specialization. This technique improves accuracy and precision of on-orbit inference by training models specialized to contexts. A context is a subset of satellite images related by a high degree of similarity in some semantically-meaningful way. For example, there is a high degree of similarity among images of forested land and among images of the ocean, while there is relatively little similarity between an image of forested land and an image of the ocean.

Context-based model specialization provides accuracy and processing time benefits. A *context-specialized* model yields higher accuracy on data from its context (see Section 5.3). Additionally, a specialized model may retain or improve accuracy while being smaller and simpler, because the model is tailored to a smaller set of data. The ability to use smaller models without degrading accuracy allows satellites to run complex legacy applications (e.g., applications designed for the datacenter cloud) with fast, simple models tailored to the inelastic space edge.

Frame tiling. This technique adjusts tile count per frame, which in turn affects the resolution of tiles input into machine inference models. A satellite frame processing application divides the image into some number of *tiles* and scales each tile to match the neural network input size. Prior work [7, 8] divides each frame into a tile count maximizing inference accuracy. For each application, Kodan sweeps a range of tile counts and selects the count maximizing downlinked data value density for the target hardware platform. When the application is not computationally-constrained on the target hardware, data value density maximizes with the most precise tiling. When the application is computationally-constrained, data value density maximizes at a less-precise tiling with lower total frame execution time (see Section 5.4).

For example, Figure 6 illustrates an application that processes a 10 000 px × 10 000 px Landsat image. The figure illustrates either dividing the image into four tiles 5000 px × 5000 px in size or 16 tiles 2500 px × 2500 px in size. After resizing each tile to the neural network input size, the processing time per tile is constant. Thus, tile count determines frame processing time; a greater number of tiles per frame increases frame processing time.

Tiling affects the quality of the sample input into a neural network. Fewer, larger tiles require more aggressive decimation to match the input dimensions of the neural network. More numerous, smaller tiles support more preservation of tile detail for neural network input.

Prior work [7] shows an empirically optimal tile count for *accuracy*, and that accuracy degrades as tile count increases or decreases from this value. In Section 5.4, we show that there also exists an empirically optimal tile count for precision, and this tile count may differ from the empirically optimal tile count for accuracy. Further, empirically optimal tile counts vary by model architecture.

Elision of processing based on sample context. This technique avoids execution of computationally-costly filtering on data samples likely to be *either* mostly high-value or mostly low-value. Instead, Kodan *discards* images from contexts containing primarily low-value data and *downlinks* images from contexts containing primarily high-value data. For example, an application searching for building footprints discards data from a context characterized by heavy cloud cover and downlinks data from a context characterized by dense urban areas. Kodan leverages elision by first identifying the tile context and then, if the tile belongs to a context with mostly high-value or low-value data, elides execution of the filtering application.

Summary: Kodan leverages three techniques to maximize data value density by adjusting application precision and execution time for each target hardware platform. Context-based model specialization improves data value density via improved precision. When an application is not computationally-constrained, all data value density improvements stem from improved precision. Sample-based processing elision skips lengthy filtering of samples belonging to contexts characterized by large amounts of high-value or low-value data. When an application is computationally-constrained, downlinking samples from a high-value context without filtering small quantities of low-value data has the effect of slightly reducing precision while significantly reducing execution time and thereby increasing data value density. Frame tiling also trades execution time and precision. When an application is not computationally-constrained, Kodan increases data value density by selecting the empirically optimal tiling for precision.

3.1 Kodan System Architecture

Kodan improves the data value density of the saturated downlink within the computational constraints of a target device by adjusting each geospatial analysis application as shown in Figure 7. A one-time application transformation step occurs before deployment. During deployment, Kodan dynamically selects these application adjustments for each sample.

Before deployment to a satellite: This one-time transformation step generates a "selection logic" that governs how, when, and which techniques (specialized models, frame tile count, and context-based elision) to deploy on orbit. Each geospatial analysis application is associated with a dataset representative of the distribution

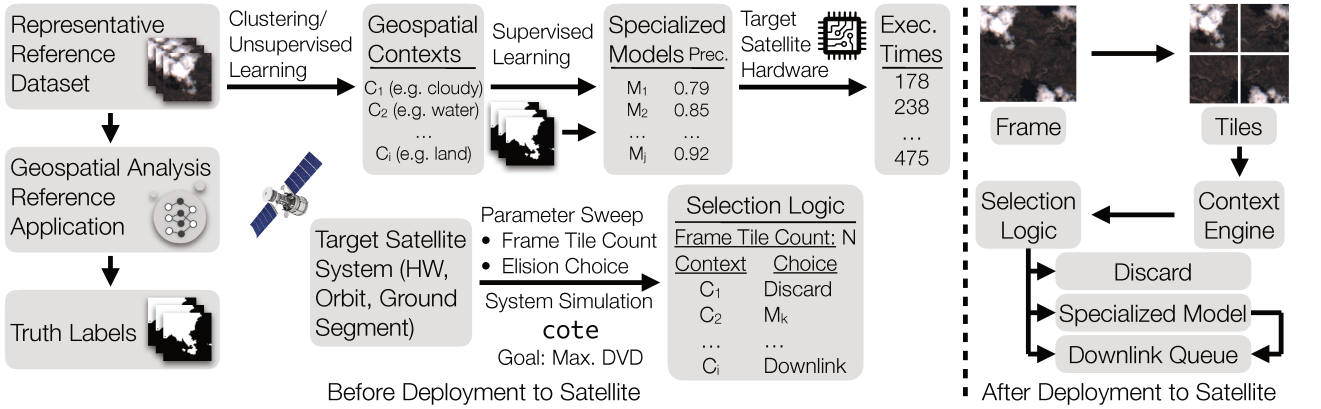

Figure 7: Left: Before deployment to a target satellite, Kodan performs a one-time transformation step. This step clusters the representative dataset into contexts and generates a selection logic for maximizing DVD. Right: After deployment to a target satellite, Kodan leverages the context engine and the selection logic to meet the soft processing deadline.

of input samples. To transform the geospatial analysis application to maximize data value density when deployed to a target satellite, Kodan (i) partitions the representative dataset into contexts; (ii) generates a context engine to classify each tile into a context, and (iii) trains and validates context-specialized models. Then, given the characteristics of the target satellite (computational capabilities, sensor and radio attributes, and orbit parameters) and its ground segment, Kodan sweeps frame tile count and context-based elision options to identify the combination of techniques maximizing data value density to produce the selection logic.

We detail context generation and the context engine in Section 3.2. To train and validate context-specialized models, Kodan adheres to machine learning best practices as described in Section 4. We detail the procedure for generating the resulting selection logic in Section 3.4.

After deployment to a satellite: During deployment, Kodan leverages the context engine and the selection logic to dynamically select application adjustments for each data sample. Based on the selected tile count per frame, Kodan splits each frame. The context engine classifies each tile into a context, which determines whether Kodan executes a specialized model or elides further processing and either downlinks or discards the sample. The satellite downlinks the application results during the next contacts with the ground segment.

3.2 Contexts and the Context Engine

Kodan must quickly and accurately classify each tile into a context for the selection logic to use when choosing application adjustments to maximize data value density within the computational constraints of the target hardware. We present two approaches for context generation and selection: expert-generated contexts and automatically-generated contexts.

Expert-generated contexts: In this approach, a subject-matter expert (SME) partitions the representative dataset into human-recognizable contexts. We observe that satellite images consist of a limited number of human-recognizable contexts that largely remain static over time. Coarse-grained examples include ocean views versus land views; more fine-grained examples include mountains, deserts, and cities.

Table 1: Per-application neural network architecture and execution times on each hardware deployment target.

Name	ML Architecture	Per-Tile Processing Time (ms) 1070 Ti	i7-7800	Orin 15W
App 1	mobilenetv2dilated-c1-deepsup	178.2	440.6	618.8
App 2	resnet18dilated-ppm-deepsup	237.6	940.6	935.6
App 3	hrnetv2-c1	321.8	1292	1515
App 4	resnet50dilated-ppm-deepsup	361.4	1787	1594
App 5	resnet50-upernet	410.9	2124	1797
App 6	resnet101-upernet	445.5	2307	1970
App 7	resnet101dilated-ppm-deepsup	475.2	2545	2040

Using this approach, a sample context can be determined from satellite position and orientation, a geographic map, and a projection of the expected satellite view onto this map. Thus, the context of each sample can be determined at the orbital edge in real time with even modest computing hardware. Because satellite orbits are highly predictable and can be modeled far in advance using tools like the `cote` simulator [8], the expert-selected context of each sample could even be pre-computed. Expert-selected contexts have the benefit of being human-explainable and quickly determined for each data sample. However, there may exist applications in which expert-selected contexts are not obvious or easy to generate.

Automatically-generated contexts: To support context generation and selection for general datasets without dependence on human experts, Kodan automatically partitions datasets into contexts. In Section 5, Kodan uses label vectors indicating the geographic and weather features present in each sample to cluster the representative dataset by similarity. Kodan creates a set of contexts using k-means clustering with these labels. During this process, Kodan sweeps cluster count and label vector distance metrics (Euclidean, Hamming, Cosine, etc.) for measuring similarity. We also consider label vector transformations, including translations, rotations, and projections

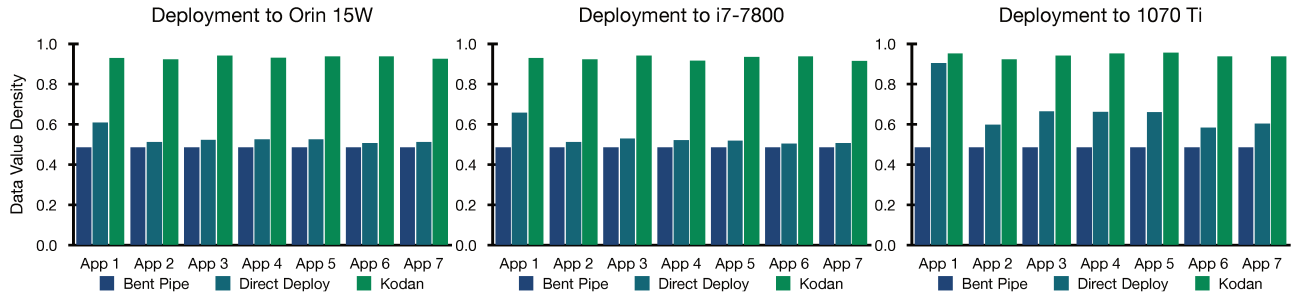

Figure 8: Kodan improves data value density compared to the bent-pipe and direct-deploy baselines.

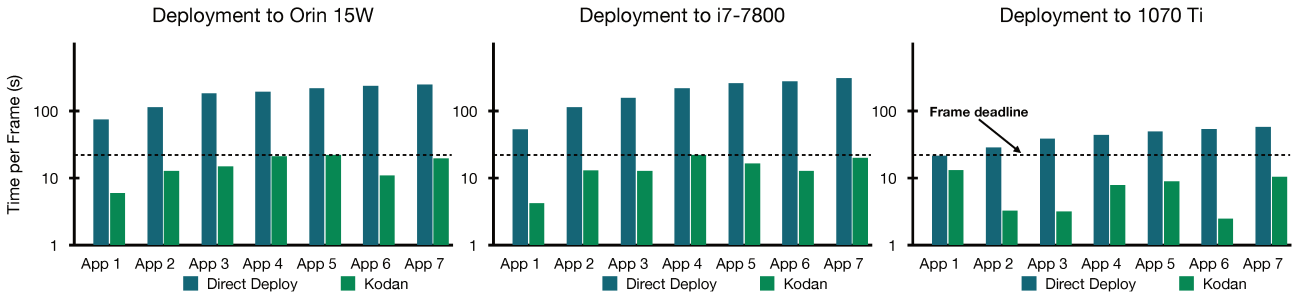

Figure 9: Kodan reduces processing time per frame by splitting each frame into fewer, larger tiles and eliding processing of tiles sorted into high-value or low-value contexts. Note the y-axis log scale.

based on per-dimension covariance properties. For automatically-generated contexts, Kodan trains a classification network to label each sample with its context — i.e., the *context engine*. The output of the deployed context engine is considered ground truth, and the resulting partition of the representative reference dataset is used to train context-specific models.

3.3 Model Specialization

Rather than execute the original reference geospatial analysis application, which is typically designed for deployment in an elastic datacenter, Kodan runs context-specialized models on target satellite hardware. After partitioning a representative dataset into contexts (Section 3.2), Kodan trains and validates models on subsets of these contexts (Section 4). Executing the original geospatial analysis application generates training labels for the representative dataset.

This process produces a set of context-specialized neural networks. Each model exhibits a known execution time on the target satellite hardware as well as known accuracy and precision characteristics across the samples sorted into its context(s) by the context engine (see Figure 7). Kodan produces models specialized for single contexts and specialized across multiple contexts, all of which are considered when generating the selection logic (see Section 3.4).

Joint generation of contexts and models. The number of contexts into which a representative dataset is partitioned is a hyperparameter affecting the number of trained, specialized models. Aside from training time during the one-time transformation step, more or fewer contexts can impact the precision and execution time benefits of context specialization. In the trivial case of a single context, the original reference geospatial application is replaced by a single neural network. As the number of contexts increases, the quantity of data samples available for training models specialized to each context decreases. As described in Section 3.2, Kodan sweeps cluster count when partitioning the representative dataset into contexts. Further exploration of this hyperparameter space represents an exciting avenue for future work.

3.4 Selection Logic

The selection logic describes a policy for deploying a combination of context-specialized models, frame tiling, and elision to maximize data value density. The best policy depends on both the downlink and computational bottlenecks. Ground station locations, satellite orbits, and radio and sensor attributes determine the downlink bottleneck. The target satellite determines the edge processing hardware and therefore the execution time of an application deployed to space. The orbit and sensors of the target satellite determine the rate of data capture and the frame deadline.

During the one-time transformation step, Kodan selects a per-frame tile count and per-context model or elision decision maximizing the data value density of the saturated downlink over the duration of the application deployment, i.e., many orbit revolutions. While the application runs on the target satellite, Kodan tiles each frame according to this selection logic. After the context engine labels a tile with a context, Kodan references the selection logic to either elide processing or determine which specialized model to use.

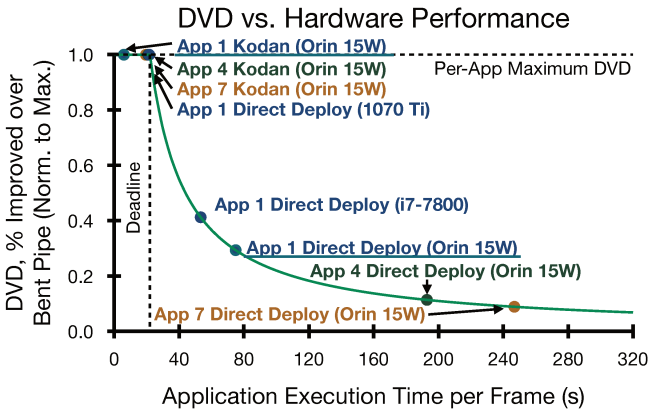

Figure 10: Each application and hardware pair exhibits a maximum data value density (DVD). Applications meeting the frame deadline achieve this maximum DVD. As application execution time increases, DVD approaches that of the bent pipe.

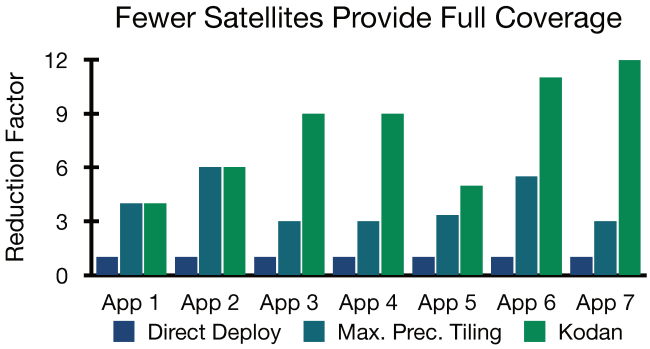

Figure 11: Using Kodan, the number of satellites in a constellation required for full ground track coverage reduces up to 12× compared to directly deploying an application to the space edge and leveraging OEC techniques in prior work [8].

Meeting the soft deadline: When the original application runs too slowly on the target satellite to meet the frame deadline on average (i.e., it experiences a computational bottleneck as in Figure 5), reducing execution time increases the data value density of the saturated downlink even when sacrificing filter precision (see Figure 10). This effect explains why a selection logic that reduces tile count per frame at the expense of filter precision, or elision of filtering for tiles from high-value contexts, improves data value density. By eliding processing that filters small amounts of low-value data from high-value contexts, more processing time is spent filtering tiles containing more significant amounts of low-value data.

Claiming idle compute time: When the application runs sufficiently fast on the target satellite to meet the frame deadline, an increase in precision — even at the cost of increased processing time — improves the data value density of the saturated downlink. This effect explains why a selection logic that increases tile count per frame at the expense of execution time can improve data value density. So long as frame processing completes by the soft deadline on average, improved filter precision ensures that a greater portion of the downlinked data consists of high-value bits. In this scenario, Kodan executes the most precise, context-specialized models that support average frame processing times less than the frame deadline. Unless elision produces more precise results than a specialized model (we observe this case when a context consists almost entirely of high-value data), the selection logic does not elect to elide processing.

4 METHODOLOGY

We evaluate Kodan with multiple space data processing applications and on multiple hardware platforms. For input, we use a Sentinel dataset [14] with classification vector labels and per-pixel masks. This dataset contains 48% high-value (i.e., non-cloudy) data and 52% low-value (i.e., cloudy) data. For test applications, we use publically-available semantic segmentation neural networks [43] customized to generate a per-pixel mask for each data sample. Table 1 summarizes these applications. We reserve a subset of the representative dataset for model validation. During training, we apply data augmentation to improve accuracy and avoid over-fitting.

We deploy each application to multiple hardware platforms. We run applications on an NVIDIA Jetson AGX Orin Tegra embedded GPU in its 15 W power mode — near the maximum reasonable power draw for a 3U cubesat subsystem. We also run applications on a Core i7-7800X CPU containing 12 cores clocked at 3.5 GHz and drawing around 140 W of power, and on a GeForce GTX 1070 Ti GPU drawing around 180 W of power. Both devices represent forward-looking computational hardware for the space edge.

Throughout our evaluation, we model orbital mechanics, data collection, and communication using the `cote` space computing simulation software [8]. We model the Landsat 8 orbit, camera sensor, and data frames by extending `cote` to import the Landsat World Reference System (WRS), and we log the image frame captures as the satellite passes over its ground track. We model the positions (latitude and longitude) and communication characteristics of the Landsat ground segment. Using `cote`, we compute the frame deadline for each satellite deployment based on its orbit characteristics.

5 EVALUATION

Our evaluation shows that, compared to directly deploying an application to a satellite or using a bent pipe, Kodan consistently and significantly improves downlink data value density across applications and hardware platforms. A key consequence of this improvement is up to a 12× reduction in satellite count to achieve full ground track coverage. Our results also provide guidance to computer architects by showing the computational capability required to meet the demands of different applications. After presenting these main results, we then focus on the individual performance improvements of each technique leveraged by Kodan. These techniques provide different benefits in different scenarios; geospatial contexts improve accuracy and precision, while both frame tiling and context-based

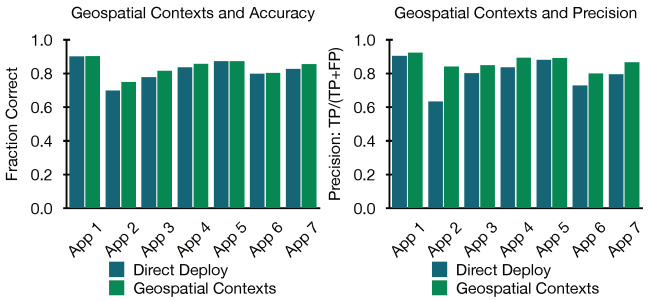

Figure 12: Left: Contexts improve accuracy (fraction of labels correct). Right: When an application meets the frame deadline, increasing precision using contexts benefits data value density.

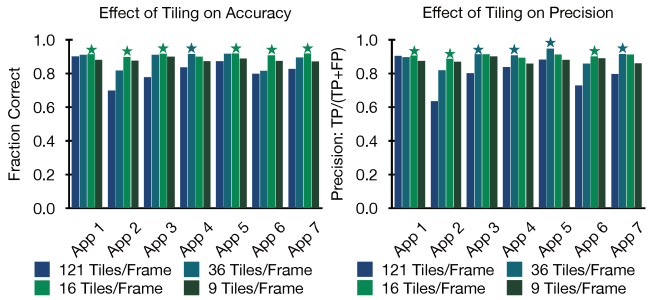

Figure 13: Left: The accuracy-maximal tiling varies by application. Right: Tiling varies precision. Absent a computational bottleneck, maximizing precision maximizes data value density.

elision balance precision and execution time. By intelligently selecting among these techniques for each application and hardware target, Kodan improves the downlink data value density across scenarios.

5.1 Kodan Improves Data Value Density

Kodan improves data value density substantially for all applications on all hardware platforms compared to a bent-pipe baseline and to a direct deployment of each application without Kodan, as shown in Figure 8. Compared to the bent-pipe baseline, Kodan improves downlink data value density between 89 and 97 percent across all applications and hardware platforms. For the bent-pipe baseline, the data value density depends solely on the prevalence of high-value data because the satellite attempts to transmit all samples to the ground. In the direct deployment case, we highlight two operating scenarios that illustrate the benefits of Kodan. In the first scenario, an application is not computationally-constrained and processes each frame within the satellite data sampling deadline. For these cases, the improvements under Kodan derive from use of geospatial contexts and context-aware model specialization to improve application accuracy and precision. In the second scenario, an application is computationally-constrained and fails to process an entire frame within the satellite data sampling deadline. For these cases, the improvements under Kodan derive from trading between precision and execution time via frame tiling and context-based elision of tile processing. Figure 9 illustrates both of these scenarios.

The data reveal several important trends. The most constrained hardware platform (the Orin operating in a 15 W mode) exhibits the greatest improvement, illustrating the benefit of trading between precision and execution time. The benefit reduces on the i7 because the applications are less computationally-constrained than on the Orin platform. On the 1070 Ti, where computational constraints are alleviated even more, the maximum precision parameter selection often maximizes data value density.

5.2 Kodan Improves Satellite Performance

Kodan reduces the required per-satellite processing capability to support the applications we evaluate, illuminating specific compute performance requirements for continuous ground track processing coverage. Recall that, for continuous ground track processing coverage, a satellite must process all tiles within the frame deadline. The bent pipe model provides a baseline data value density by indiscriminately transmitting observations until saturating the downlink. For each application, the maximum data value density depends on the highest precision the application achieves within the frame deadline.

In Figure 10, we evaluate the relationship between frame processing time (i.e., compute performance) and improvement in data value density compared to the bent pipe minimum. The plot shows that reducing frame processing time improves data value density until the time to process all tiles is less than the frame deadline. Once frame processing time drops below the deadline, the downlink data value density of an application is limited by its precision.

Points in the plot are measured data value densities for different application/hardware combinations from our evaluation. These points show Kodan improves data value density for Applications 1, 4, and 7, especially when exceeding the frame deadline when directly deployed to the hardware targets. Each application consists of a different number of operations, and therefore they require different amounts of time to complete; Application 1 consists of the fewest number of operations and Application 7 consists of the most. Application 1 directly deployed to the 1070 Ti meets the deadline; thus, the downlink data value density depends only on the application precision (see Figure 8 for the effects of precision differences between direct deploy and Kodan on data value density). When directly deployed to the i7 or to the Orin, Application 1 falls short of the maximum data value density because it executes too slowly to meet the deadline. Using Kodan, Application 1 executes significantly faster. The resulting idle time affords Kodan the opportunity to deploy models of higher precision to improve data value density.

Directly deploying Application 4 or Application 7 to the Orin gives worse results. Using Kodan, both applications attain their maximum achievable data value density due to significant reductions in execution time. Future satellites — especially nanosatellites — are likely to include hardware similar to the Orin due to its low power consumption. Rather than directly deploying to such a device, using Kodan alleviates the computational bottleneck, which exists even for the simplest of our benchmark applications.

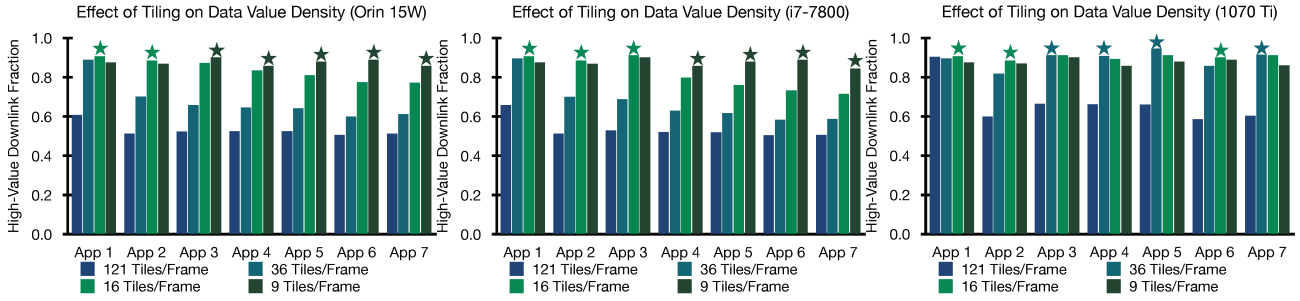

Figure 14: On more computationally-constrained platforms like the Orin, more aggressive tiling (nine per frame) maximizes data value density. As the computational bottleneck eases (1070 Ti) tilings that maximize precision also maximize data value density.

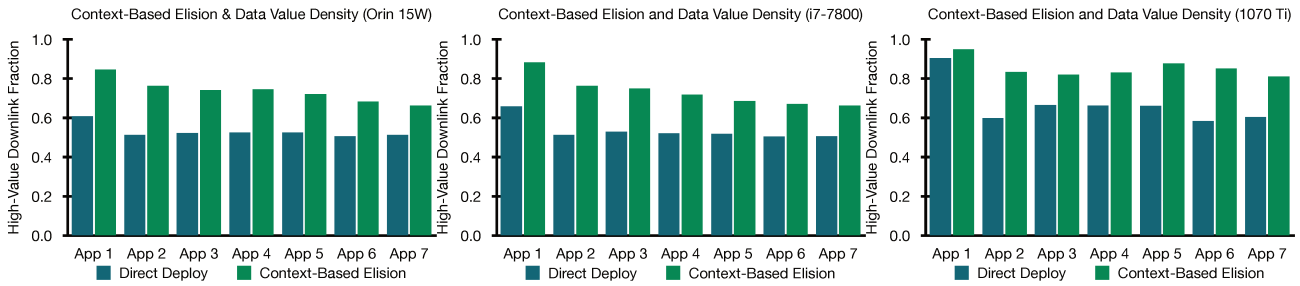

Figure 15: Some contexts consist mostly of high-value or low-value data. Downlinking samples from high-value contexts and discarding samples from low-value contexts — i.e., eliding further processing — eases the computational bottleneck by freeing time to filter samples from other contexts.

5.3 Contexts Improve Precision

Kodan uses geospatial contexts to improve both accuracy and precision for all applications, garnering a data value density benefit regardless of whether the satellite experiences a computational bottleneck. With a saturated downlink, precision — here, the fraction of correctly labeled pixels — determines data value density, because incorrectly-labeled pixels "pollute" the already-saturated downlink. Figure 12, right, displays the effect of context-specific model selection on application precision. The increase in application precision with geospatial contexts is more pronounced than the increase in application accuracy. For example, Application 2 improves in precision by 33% with geospatial contexts. Figure 12, left, shows that context-specific model selection also improves accuracy by up to 7.5%. Applications with the lowest baseline accuracy experience the greatest improvement (e.g., Application 2).

5.4 Tiling Trades Precision and Performance

Kodan uses frame tiling to improve data value density by adjusting tile count per frame to balance frame processing time with application precision. Figure 13, left, shows how tiling affects application accuracy. Figure 13, right, displays the effect of tiling on application precision. Each application has a tile count that maximizes accuracy and a tile count that maximizes precision. Increasing or decreasing tile count from these optimal tilings decreases accuracy or precision; this finding is consistent with prior work [7]. Absent a computational bottleneck, the maximum precision tiling yields the maximum data value density.

When computationally bottlenecked, Kodan selects a tiling to reduce execution time at the expense of precision in order to increase data value density. Figure 14 illustrates this effect. Tiling provides up to a 50% improvement in data value density for the most computationally-constrained platform and the most computationally-costly application (i.e., Application 7 on the Orin mobile GPU). To understand this result, recall that processing more data provides substantial data value density improvements until meeting the frame deadline, when *precisely* processing tiles becomes most important (see Figure 10). For simpler applications (e.g., Application 1) and more capable platforms (e.g., the 1070 Ti), less aggressive tiling supports higher precision within the frame deadline.

5.5 Elision Improves Data Value Density

Kodan uses context-based elision to downlink or discard samples from primarily high-value or low-value contexts and thereby substantially increases data value density. Figure 15 illustrates this effect. Improvements from elision are more pronounced under a greater computational bottleneck; for example, when Application 7 deploys to the Orin. As the computational bottleneck eases (e.g., simpler applications or better hardware), the benefit of elision becomes less pronounced because the system processes more samples within the deadline. For example, the data value density improvement for Application 1 is 39% on the Orin platform, while the data value

density improvement for Application 1 is 34% on the i7 platform. On the 1070 Ti platform, Application 1 is not computationally bottlenecked; instead, the benefits derive from the precision improvement of sending down *likely* high-value pixels without processing, rather than imprecisely processing those pixels and transmitting a worse result.

6 RELATED WORK

This work spans computer systems, space systems, space networks, and systems ML. Section 2 provides an overview of the emerging computational space systems domain, and Section 2.1 characterizes major challenges of this research area. Recent works in orbital edge computing provide context for this work. The space networking challenge of the bent pipe bottleneck is quantified in [7], which also introduces the concept of a computational satellite deploying machine inference to address this challenge. Orbital edge computing [8] aims to address the downlink bottleneck and introduces the computational bottleneck of the inelastic space edge. Challenges and opportunities in this domain are examined in [30].

The computational bottleneck poses a major space systems challenge. Although monolithic satellites cost hundreds of millions of US dollars each, this high cost does not provide high-performance onboard processors. These systems must operate for decades to justify the high cost, which means that computer hardware must be low-risk and highly-reliable. Often, these systems use decades-old, "flight heritage" CPUs. After two or more decades of operation, onboard processors could be nearly half a century behind the state-of-the-art.

For example, LEON processors — which receive significant support from the European Space Agency (ESA) — implement the 32-bit SPARC V8 instruction set architecture (ISA). A recent implementation achieved a clock frequency of 250 MHz [1]. The EO-1 makes use of a 12 MHz Mongoose-V central processing unit (CPU), which implements the 32-bit MIPS ISA, to demonstrate autonomous science [6]. The RAD5500 implements a 64-bit PowerPC ISA operating at 466 MHz [2]. The limited performance of these CPUs stems from larger technology nodes and functional unit duplication, which help to provide reliability in the space environment. Extremely high costs demand extremely low risk and long-duration missions at the expense of performance. Recent trends in space systems have started to consider COTS embedded computer systems [28, 29].

Several works consider models optimized for accuracy or speed in terrestrial applications [4, 21, 22, 34, 37, 41]. Works on embedded, terrestrial, wireless sensor systems study the tradeoff between computation and communication for energy-harvesting devices [10, 11, 16, 17, 35]. These works identify the relatively high energy cost of communication and quantify benefits to spending energy on computation instead. While these terrestrial systems can transmit data at any time within energy constraints, satellites can transmit data only while near a ground station. In this work, we focus on model specialization for geospatial contexts at the orbital edge to improve the data value density of a saturated downlink.

Space networking is a growing field of research [3, 19, 24]. Much work focuses on inter-satellite communication, which is a challenging engineering question encompassing control theory, orbital dynamics, robotics, and energy-performance tradeoffs. Less attention has been paid to the challenges of the downlink bottleneck and the saturated downlink, which we examine in this work. Alternate approaches to addressing the downlink bottleneck [39] are complementary to this work by enabling higher performance with even smaller satellite constellation populations.

7 CONCLUSION AND FUTURE WORK

The increasing accessibility of space opens the orbital edge to new geospatial analysis applications. A limited downlink creates a need for on-orbit processing to extract value from sensor data. However, constraints on orbital edge computing limit the value of satellite-based applications. Kodan mitigates the downlink bottleneck and the computational bottleneck for space edge systems by leveraging geospatial contexts and specializing satellite computation to balance application processing time with precision. This approach contrasts with expensive, constellation-oriented techniques that extract value from data but require many satellites to do so. We implement and evaluate Kodan, which increases the density of valuable data downlinked from LEO between 89 and 97 percent without changing ground infrastructure or radio attributes despite bottlenecked bandwidth and computing.

Orbital edge computing has an exciting future with many open research questions. Building on this work, computational space system designers should improve sensor coverage and computing capability of constellations through co-design of system-level optimizations and computer architecture while avoiding unfavorable cost-scaling of high device counts. Future constellations will feature heterogeneous sensors, computational capabilities, and actuators. Some satellites may share data via crosslinks and distribute processing so that each satellite need not contain telescope optics, precision pointing, laser communication, and a high-end GPU. Instead, a heterogeneous constellation supports *hardware* specialization (as opposed to processing specialization based on sample context), allowing individual satellites to contain fewer subsystems are therefore be more simple. Energy constraints, orbital dynamics, and client mission goals place time-dependent constraints on satellite operations. Communication and computation will remain perennial challenges for space-based computer systems. Kodan demonstrates that these challenges are surmountable with new techniques tailored to the unique constraints of the orbital edge.

REFERENCES

[1] Jan Andersson, Magnus Hjorth, Fredrik Johansson, and Sandi Habinc. 2017. Leon processor devices for space missions: First 20 years of leon in space. In *SMC-IT*. IEEE.
[2] Richard Berger, Steve Chadwick, Ernesto Chan, Richard Ferguson, Patrick Fleming, Jane Gilliam, Michael Graziano, Mary Hanley, Andrew Kelly, Marla Lassa, et al. 2015. Quad-core radiation-hardened system-on-chip power architecture processor. In *Aerospace Conference*. IEEE.
[3] Debopam Bhattacherjee, Waqar Aqeel, Ilker Nadi Bozkurt, Anthony Aguirre, Balakrishnan Chandrasekaran, P Brighten Godfrey, Gregory Laughlin, Bruce Maggs, and Ankit Singla. 2018. Gearing up for the 21st century space race. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*.
[4] Jiashen Cao, Ramyad Hadidi, Joy Arulraj, and Hyesoon Kim. 2021. Thia: Accelerating video analytics using early inference and fine-grained query planning. *arXiv preprint arXiv:2102.08481* (2021).
[5] Jeroen Cappaert. 2018. Building deploying and operating a cubesat constellation-exploring the less obvious reasons space is hard. In *Proc. AIAA/USU Conf. Small Satellites*.

[6] Steve Chien, Rob Sherwood, Daniel Tran, Benjamin Cichy, Gregg Rabideau, Rebecca Castano, Ashley Davies, Rachel Lee, Dan Mandl, Stuart Frye, et al. 2004. The EO-1 autonomous science agent. In *Joint Conference on Autonomous Agents and Multiagent Systems*.

[7] Bradley Denby and Brandon Lucia. 2019. Orbital edge computing: Machine inference in space. *IEEE Computer Architecture Letters* (2019).

[8] Bradley Denby and Brandon Lucia. 2020. Orbital edge computing: Nanosatellite constellations as a new class of computer system. In *Architectural Support for Programming Languages and Operating Systems*.

[9] Bradley Denby, Emily Ruppel, Vaibhav Singh, Shize Che, Chad Taylor, Fayyaz Zaidi, Swarun Kumar, Zac Manchester, and Brandon Lucia. 2022. Tartan Artibeus: A Batteryless, Computational Satellite Research Platform. (2022).

[10] Harsh Desai and Brandon Lucia. 2020. A power-aware heterogeneous architecture scaling model for energy-harvesting computers. *Computer Architecture Letters* (2020).

[11] Harsh Desai, Matteo Nardello, Davide Brunelli, and Brandon Lucia. 2022. Camaroptera: A long-range image sensor with local inference for remote sensing applications. *Transactions on Embedded Computing Systems (TECS)* (2022).

[12] Kiruthika Devaraj, Ryan Kingsbury, Matt Ligon, Joseph Breu, Vivek Vittaldev, Bryan Klofas, Patrick Yeon, and Kyle Colton. 2017. Dove High Speed Downlink System. In *Proc. AIAA/USU Conf. Small Satellites*.

[13] Warren Ferster. 2012. DigitalGlobe Adding Infrared Capability to WorldView-3 Satellite. *Space News, https://spacenews.com/digitalglobe-adding-infrared-capability-worldview-3-satellite/* (2012).

[14] Alistair Francis, John Mrziglod, Panagiotis Sidiropoulos, and Jan-Peter Muller. 2020. Sentinel-2 Cloud Mask Catalogue. https://doi.org/10.5281/zenodo.4172871.

[15] Warren Frick and Carlos Niederstrasser. 2018. Small Launch Vehicles-A 2018 State of the Industry Survey. In *Proc. AIAA/USU Conf. Small Satellites*.

[16] Graham Gobieski, Nathan Beckmann, and Brandon Lucia. 2018. Intermittent deep neural network inference. In *SysML Conference*.

[17] Graham Gobieski, Brandon Lucia, and Nathan Beckmann. 2019. Intelligence beyond the edge: Inference on intermittent embedded systems. In *Architectural Support for Programming Languages and Operating Systems*.

[18] Ritwik Gupta, Richard Hosfelt, Sandra Sajeev, Nirav Patel, Bryce Goodman, Jigar Doshi, Eric Heim, Howie Choset, and Matthew Gaston. 2019. xBD: A Dataset for Assessing Building Damage from Satellite Imagery. *arXiv preprint arXiv:1911.09296* (2019).

[19] Mark Handley. 2018. Delay is not an option: Low latency routing in space. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*.

[20] William Harwood. 2013. NASA launches USD 855 million Landsat mission. *CBS News, https://www.cbsnews.com/news/nasa-launches-855-million-landsat-mission/* (2013).

[21] Nada Ibrahim, Preeti Maurya, Omid Jafari, and Parth Nagarkar. 2021. A survey of performance optimization in neural network-based video analytics systems. *arXiv preprint arXiv:2105.14195* (2021).

[22] Daniel Kang, John Emmons, Firas Abuzaid, Peter Bailis, and Matei Zaharia. 2017. Noscope: optimizing neural network queries over video at scale. *arXiv preprint arXiv:1703.02529* (2017).

[23] Michael D King, Steven Platnick, W Paul Menzel, Steven A Ackerman, and Paul A Hubanks. 2013. Spatial and temporal distribution of clouds observed by MODIS onboard the Terra and Aqua satellites. *Transactions on Geoscience and Remote Sensing* (2013).

[24] Tobias Klenze, Giacomo Giuliari, Christos Pappas, Adrian Perrig, and David Basin. 2018. Networking in Heaven as on Earth. In *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*.

[25] Wiley J Larson and James Richard Wertz. 1992. *Space mission analysis and design*. Microcosm.

[26] Lawrence Leung, Vincent Beukelaers, Simone Chesi, Hyosang Yoon, Daniel Walker, and Joshua Egbert. 2018. ADCS at scale: Calibrating and monitoring the dove constellation. In *Proc. AIAA/USU Conf. Small Satellites*.

[27] Thomas R Loveland and James R Irons. 2016. Landsat 8: The plans, the reality, and the legacy. *Remote Sensing of Environment* (2016).

[28] Tyler M Lovelly and Alan D George. 2017. Comparative analysis of present and future space-grade processors with device metrics. *Journal of aerospace information systems* (2017).

[29] Tyler M Lovelly, Travis W Wise, Shaun H Holtzman, and Alan D George. 2018. Benchmarking analysis of space-grade central processing units and field-programmable gate arrays. *Journal of Aerospace Information Systems* (2018).

[30] Brandon Lucia, Brad Denby, Zachary Manchester, Harsh Desai, Emily Ruppel, and Alexei Colin. 2021. Computational Nanosatellite Constellations: Opportunities and Challenges. *GetMobile: Mobile Computing and Communications* (2021).

[31] James McClain. 2021. The Azavea Cloud Dataset. https://github.com/azavea/cloud-model.

[32] Arash Mehrparvar, D Pignatelli, J Carnahan, R Munakat, W Lan, A Toorian, A Hutputanasin, and S Lee. 2014. *CubeSat Design Specification Rev. 13*. Technical Report. California Polytechnic State University, San Luis Obispo.

[33] Elizabeth M Middleton, Stephen G Ungar, Daniel J Mandl, Lawrence Ong, Stuart W Frye, Petya E Campbell, David R Landis, Joseph P Young, and Nathan H Pollack. 2013. The earth observing one (EO-1) satellite mission: Over a decade in space. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* (2013).

[34] Mahyar Najibi, Bharat Singh, and Larry S Davis. 2019. Autofocus: Efficient multi-scale inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*.

[35] Matteo Nardello, Harsh Desai, Davide Brunelli, and Brandon Lucia. 2019. Camaroptera: A batteryless long-range remote visual sensing system. In *Workshop on Energy Harvesting and Energy-Neutral Sensing Systems*.

[36] S Radu, M Uludag, S Speretta, J Bouwmeester, A Dunn, T Walkinshaw, P Kaled Da Cas, and C Cappelletti. 2018. *The PocketQube Standard Issue 1*. Technical Report. TU Delft.

[37] Abhijit Suprem, Joy Arulraj, Calton Pu, and Joao Ferreira. 2020. Odin: automated drift detection and recovery in video analytics. *arXiv preprint arXiv:2009.05440* (2020).

[38] U.S. Geological Survey. 2021. Landsat Path/Row World Reference System. https://www.usgs.gov/landsat-missions/landsat-shapefiles-and-kml-files.

[39] Deepak Vasisht and Ranveer Chandra. 2020. A distributed and hybrid ground station network for low earth orbit satellites. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks*.

[40] Deepak Vasisht, Zerina Kapetanovic, Jongho Won, Xinxin Jin, Ranveer Chandra, Sudipta Sinha, Ashish Kapoor, Madhusudhan Sudarshan, and Sean Stratman. 2017. FarmBeats: An IoT Platform for Data-Driven Agriculture. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*.

[41] Ran Xu, Rakesh Kumar, Pengcheng Wang, Peter Bai, Ganga Meghanath, Somali Chaterji, Subrata Mitra, and Saurabh Bagchi. 2021. ApproxNet: Content and contention-aware video object classification system for embedded clients. *ACM Transactions on Sensor Networks (TOSN)* (2021).

[42] Zac Manchester. 2015. KickSat. http://zacinaction.github.io/kicksat/.

[43] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. 2017. Scene Parsing through ADE20K Dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Received 2022-10-20; accepted 2023-01-19