

ATLAS: Expert-Guided Sustainable Satellite Backhaul for Off-Grid Wildlife Monitoring

Hao Fang

Simon Fraser University
AXSOTA Intelligence Inc.
Burnaby, BC, Canada
fanghaof@sfu.ca

Miao Zhang*

Simon Fraser University
AXSOTA Intelligence Inc.
Burnaby, BC, Canada
mza94@sfu.ca

Haoyuan Zhao

Simon Fraser University
Burnaby, BC, Canada
hza127@sfu.ca

Long Chen

Simon Fraser University
Burnaby, BC, Canada
longchen.cs@ieee.org

Yi Ching Chou

Simon Fraser University
Burnaby, BC, Canada
ycchou@sfu.ca

Chi Xu

Simon Fraser University
AXSOTA Intelligence Inc.
Burnaby, BC, Canada
chix@sfu.ca

Cheng Pan

Simon Fraser University
Burnaby, BC, Canada
chengpan@ieee.org

Jiangchuan Liu

Simon Fraser University
Burnaby, BC, Canada
jcliu@sfu.ca

Abstract

Remote wildlife monitoring increasingly depends on high-fidelity cameras and sensors deployed far from grid power and terrestrial networks. Low Earth orbit satellite links can provide connectivity in these settings, but their high power draw and variable bandwidth, combined with intermittent on-site solar harvesting, make continuous backhaul infeasible. Current systems therefore filter at the edge, with motion triggers that waste limited energy and bandwidth on environmental artifacts, or fixed vision models that cannot follow evolving expert objectives.

We present ATLAS, an expert-guided edge-to-cloud system that treats satellite backhaul as a scarce resource for off-grid wildlife monitoring. ATLAS uses foundation models to compile natural-language expert intent into semantic scoring logic, combines semantic relevance with environmental and energy state, and uses a constrained reinforcement-learning scheduler to decide which clips to upload and when. As a representative case study, we deployed ATLAS for over six months across 18 off-grid salmon-monitoring river sites. ATLAS achieves a mean clip-selection accuracy of 87.4%, exceeding mean baseline accuracy by 26.4 percentage points, while maintaining near-continuous operation through low-solar periods in which baselines incur 16.0% downtime.

CCS Concepts: • **Computer systems organization** → **Embedded and cyber-physical systems**; **Sensor networks**; • **Computing methodologies** → *Natural language processing*; • **Applied computing** → **Environmental sciences**.

*Corresponding author.



This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

SIGOPS ATC 2026, Hong Kong, China

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2818-1/2026/11

<https://doi.org/10.1145/3828523.3857049>

Keywords: LEO satellite networks, foundation models, energy harvesting, edge computing, wildlife monitoring

ACM Reference Format:

Hao Fang, Miao Zhang, Haoyuan Zhao, Long Chen, Yi Ching Chou, Chi Xu, Cheng Pan, and Jiangchuan Liu. 2026. ATLAS: Expert-Guided Sustainable Satellite Backhaul for Off-Grid Wildlife Monitoring. In *The 2026 ACM SIGOPS Annual Technical Conference (SIGOPS ATC 2026)*, November 15–18, 2026, Hong Kong, China. ACM, New York, NY, USA, 15 pages. <https://doi.org/10.1145/3828523.3857049>

1 Introduction

Remote wildlife monitoring is a cornerstone of conservation and ecosystem management [6, 17]. Modern sensors such as high-definition video traps and continuous acoustic recorders produce large volumes of high-fidelity data for the domain experts who run these programs. Until recently, off-grid sites had no way to move that data off-site in time for experts to act on it.

The recent proliferation of low Earth orbit (LEO) satellite constellations addresses this gap, offering global Internet access and low-latency communication from the edge [3, 27, 34, 37]. Even with ubiquitous connectivity, off-grid deployments remain bounded by energy [16, 39], since they rely on unstable solar harvesting while the LEO user terminal (UT) draws tens of watts when transmitting, far more than the camera and edge computer it serves (§2.2). During the most severe overcast periods at one of our rainforest sites, daily harvest fell to 21 Wh, about a tenth of what a standard outdoor camera needs for a day and enough for less than 20 minutes of active transmission on the terminal.

A long line of work on energy-constrained sensing and communication addresses two questions: what to transmit and when. Motion triggers [49] and on-device computer vision (CV) models [6, 22, 58] decide which footage is captured or sent. Harvest-aware runtimes adapt duty cycles to predicted energy [18, 39], and energy-aware schedulers send data only when it costs less energy or carries more information [35, 51]. These mechanisms fall short in two respects.



Figure 1. (a) Reaching deployment sites often requires multiple modes of transportation, including boats and seaplanes, making site visits costly. (b, c) A typical remote wildlife monitoring deployment in a temperate rainforest, with solar panels and the Starlink user terminal (UT) mounted in the tree canopy to maximize sunlight exposure and minimize signal obstruction, respectively. (d) Edge system hardware, including the edge computing enclosure and battery pack.

First, neither reflects expert intent, since content selection encodes relevance as classes fixed before deployment and transmission scheduling values data by a generic measure of information. Second, each answers only its own question, so content is largely selected without regard to the energy state, and transmissions are scheduled without regard to what the content shows. A monitoring system must therefore decide jointly, based on expert intent, which video clips are worth their energy cost and when to spend it.

Optimizing these decisions first requires understanding what information is actually valuable to domain experts. Through our partnerships with wildlife monitoring organizations, particularly fisheries teams, we find that expert intent is rarely reducible to a small set of static labels. Instead, interests are often expressed as complex, context-dependent tasks articulated in natural language, such as “identify upstream movement of adult salmon during peak return windows” or “flag unusual species appearances following heavy rainfall events.” Moreover, these objectives are dynamic. Experts may focus on different species across seasons, shifting priority from migration onset to behavioural anomalies or size distributions. Complicating this further, the visual environment is constantly changing. Water turbidity, lighting, and vegetation can shift rapidly, meaning the same high-value event can look markedly different under local conditions.

Foundation models (FMs), particularly large language and multimodal models, offer a way to translate such intent into system behaviour [32, 48, 56]. Unlike supervised models restricted to fixed classes, they generalize beyond those classes and follow instructions in natural language [46, 50, 57, 62], so that expert knowledge can be supplied in the same form and updated as conservation goals and environmental conditions evolve. However, FMs address only what to transmit and cannot weigh a clip’s value to the experts against its energy cost.

We present ATLAS, an energy-aware, expert-guided edge-to-cloud system for off-grid wildlife monitoring. ATLAS treats satellite connectivity as a first-class, budgeted resource

and reasons jointly about what a clip is worth to the experts and about the constraints imposed by variable solar energy, high uplink power draw, and time-varying bandwidth. A fine-tuned language model, the Semantic Assessment Agent, reads the compiled expert intent, site-specific knowledge, and frame-level descriptions of each candidate clip, and assigns the clip a relevance score and an uncertainty estimate. All scored candidates enter a local event buffer. The Adaptive Uplink Scheduler combines these assessments with energy telemetry, forecasts, and link measurements, and its reinforcement learning (RL) policy decides *which* clips to upload and *when*, balancing expert relevance, timeliness, and battery reserves. Expert feedback updates the intent and knowledge that the agent reads, so that ATLAS follows evolving goals without retraining.

As a representative case study, we partnered with North American fisheries organizations focused on salmon conservation and deployed ATLAS at 18 off-grid river sites along the Pacific coast. Over a deployment of more than six months spanning the entire salmon migration and spawning season, our evaluation shows that ATLAS effectively interprets experts’ complex intent and incorporates expert knowledge to identify video clips that best match expert interests, thereby reducing unnecessary uplink energy expenditure. Compared with representative baselines, ATLAS identifies expert-relevant video clips with 87.4% accuracy, achieving an average improvement of 26.4 percentage points. In addition, ATLAS maintains near-continuous operation throughout extended low-solar periods by proactively budgeting energy for communication, whereas the baselines experience an average system downtime of 16.0% due to energy depletion. These results demonstrate that expert-guided, energy-aware uplink control can substantially improve both observational utility and long-term system sustainability. Our primary contributions are summarized as follows:

- We design ATLAS, an edge-to-cloud wildlife monitoring system that treats satellite connectivity as a scarce,

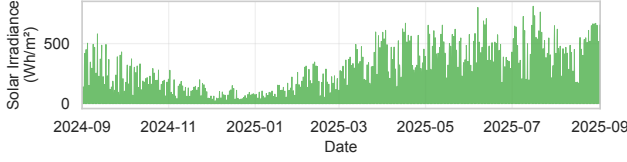


Figure 2. Measured solar irradiance over one year. Higher values correspond to increased solar energy availability, directly impacting battery charging and system uptime.

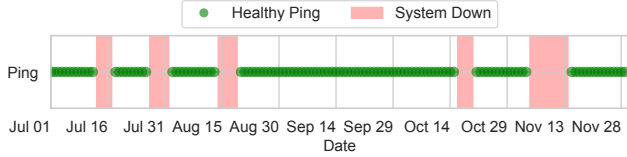


Figure 3. Preliminary system uptime measurements at a representative riverside site on the west coast of North America.

budgeted resource, and decides jointly which video clips are worth transmitting and when.

- We build a Semantic Assessment Agent that scores each clip against compiled expert intent and site knowledge, with an uncertainty estimate, and an expert-in-the-loop workflow that updates the intent and knowledge it reads without retraining.
- We formulate uplink scheduling as a constrained Markov decision process that balances semantic utility, information freshness, and battery reserves under stochastic solar harvesting and variable satellite connectivity, with a runtime safety shield that enforces per-step energy limits.
- We fully implement ATLAS on off-grid hardware and validate it through real field deployments and trace-driven experiments. Our evaluation demonstrates that ATLAS improves both clip-selection accuracy and system uptime over three representative baselines.

2 Motivation

Our deployments face constraints long studied in energy-harvesting and delay-tolerant sensing systems [5, 7, 16–18, 39]. Energy harvest varies with weather and season, installations operate unattended for months, and outages leave permanent gaps in the ecological record. Drawing on observations from our sites, this section examines how these constraints manifest in a LEO-connected video system and identifies the resulting design requirements.

2.1 Challenges of Wild Deployment Environments

Wildlife monitoring is increasingly conducted in remote, ecologically sensitive habitats, including tropical and temperate rainforests, high-latitude wetlands, alpine regions, and protected river systems, where reliable grid power and terrestrial Internet connectivity are often unavailable. Our monitoring

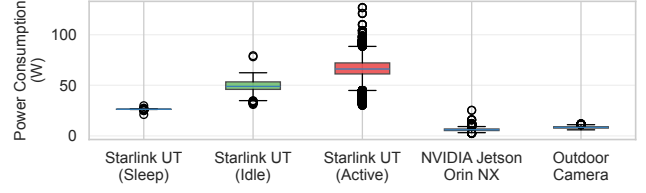


Figure 4. Measured power draw of on-site components, including the Starlink Standard (Gen 3) user terminal (UT) in sleep, idle, and active states.

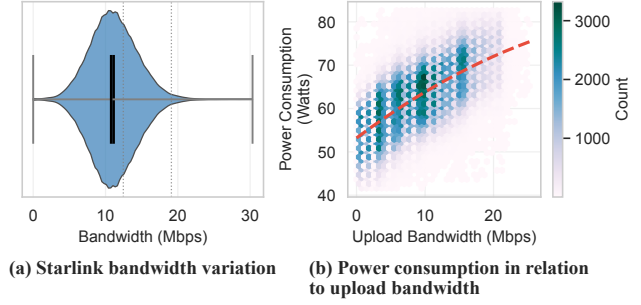


Figure 5. Measured Starlink UT upload bandwidth and its effect on power draw.

sites along salmon-bearing rivers on the Pacific coast of North America exemplify these conditions. As shown in Figure 1, each site operates entirely off-grid, relying on solar energy harvesting for power and a satellite link for Internet connectivity. This reliance on solar power makes energy availability dependent on local weather conditions.

Figure 2 presents one year of irradiance measurements from a temperate rainforest site. Solar energy availability is substantially higher in summer than in winter and exhibits pronounced day-to-day variability within each season. Figure 3 illustrates the operational consequences of this variability. During a preliminary deployment at a riverside site, five episodes of adverse weather led to battery depletion and system shutdowns, with a median downtime of 120 hours. Recovery from each shutdown required manual intervention. An upload that appears feasible given the current battery state of charge may therefore lead to a shutdown several days later. Uplink energy must be budgeted over multi-day horizons using energy-harvest forecasts.

2.2 Power and Bandwidth Constraints of Satellite Communication

At these sites, satellite communication is the only means of transferring data off-site. Today, LEO services such as Starlink provide connectivity to these remote sites at a reasonable cost, but for an off-grid system, the link imposes two further constraints: a high, bursty power draw and a limited, variable uplink bandwidth.

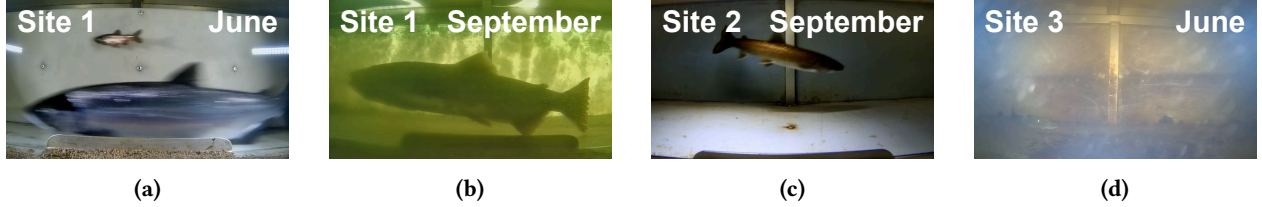


Figure 6. Visual variability in salmon footage, showing seasonal change at one site and differing conditions across sites.

Figure 4 shows the measured power draw of the on-site components. The Starlink UT consumes substantially more power than any other component, even in sleep and idle states, whereas the edge computer and camera typically draw less than 10 W. In addition, waking the terminal and establishing a usable link takes 183 s at 50.6 W, corresponding to a fixed energy cost of approximately 9.26 kJ, independent of the volume of data transmitted. These costs limit the number of hours per day for which the terminal can be kept awake within the available energy budget. Uploads must therefore be batched to amortize the wake-up cost rather than transmitted as individual events occur.

Figure 5(a) shows the measured uplink throughput, which ranges from 0 to 30.33 Mbps with a median of 10.78 Mbps and varies with weather conditions, terrain occlusion, and antenna orientation. Power draw also rises with throughput (Figure 5(b)). This variability makes the transmission time and energy required to deliver a clip uncertain in advance. Upload batches must be sized according to forecasts of available uplink bandwidth.

Together, these constraints create a trade-off between timely data delivery and sustained system operation. Uploading more supports downstream analysis but consumes energy and link time, whereas uploading less conserves energy but can delay critical information. Compression helps only up to a point, because a clip reduced below the fidelity that experts need to verify it loses its value. A keyframe set, for example, cannot show movement when the task depends on it. Which clips to upload, how much to send in each window, and when to wake the terminal must therefore be planned jointly, using forecasts of both available bandwidth and energy harvest.

2.3 Concept Drift and the Limits of Static Models

Wildlife monitoring systems operate in dynamic and heterogeneous environments, so a single model trained on data from a limited set of conditions rarely generalizes reliably. Models that perform well in one river, season, or lighting condition may degrade significantly when applied to another [2]. Long-term deployment amplifies the problem, since the data distribution drifts away from the training distribution as the environment evolves, and performance degrades over time.

Wildlife monitoring also presents intrinsic challenges from species diversity and visual ambiguity. Many species, such

as salmonids, differ only subtly from one another, and individuals of the same species vary in appearance, so they are difficult to distinguish from appearance alone, especially under noisy sensing conditions. Figure 6 shows salmon species that appear visually similar alongside footage captured under diverse water conditions. Trained experts resolve these distinctions by drawing on contextual cues such as seasonality, behaviour, and migration patterns, which conventional CV models cannot capture from raw imagery alone.

Much of this expert knowledge is not encoded in labelled datasets. It exists in unstructured forms such as field notes, written reports, regulatory guidelines, and oral knowledge passed between practitioners, for example “count only upstream-moving adults,” “ignore glare-induced false positives,” or “flag unusually pale fish during low flow.” Detection and classification models such as YOLO-style architectures [33] are trained from visual supervision over a fixed label set and offer no way to incorporate such rules. A monitoring system must take expert knowledge in the form in which it exists, as natural-language rules and descriptions, and revise what counts as relevant as that knowledge changes, without retraining or large-scale labelling.

3 ATLAS Overview

ATLAS uses expert-guided semantic assessment to identify relevant observations from remote, off-grid deployments. Its energy-aware scheduler determines which observations to upload over the satellite link and when, given the available energy and bandwidth. Figure 7 illustrates the end-to-end workflow of ATLAS.

Low-power sensing runs continuously on the device. A lightweight Activity Detector watches the camera feed for image-based motion cues, and the Dynamic Sampler adapts the frame rate to its output. While no activity is detected, the sampler keeps a low frame rate to minimize computation and storage. When activity is detected, it raises the rate and extracts a short clip that preserves enough visual detail for analysis. Each clip forms a candidate event, the unit that the rest of the system assesses and schedules, so that later stages spend energy only on footage that contains activity.

For each candidate event, the **Domain Vision Model (DVM)**, fine-tuned on footage from our deployment sites, labels and tracks objects in the clip, and the **Vision Language Model (VLM)** adds a holistic description of each frame when

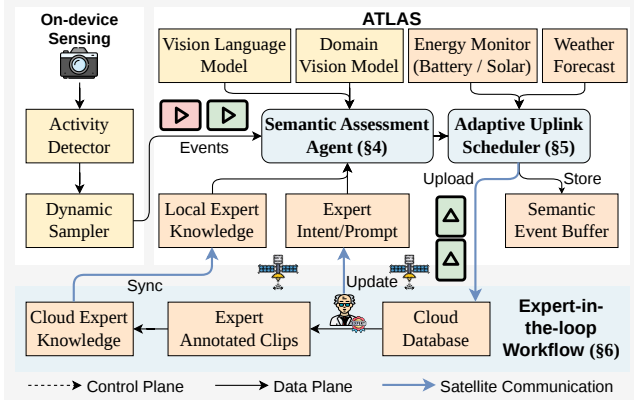


Figure 7. ATLAS system overview

the expert intent requires such context. The **Semantic Assessment Agent** reads these frame-level descriptions together with the compiled expert intent and the Local Expert Knowledge, which stores the site’s expert knowledge and event memory for retrieval. It outputs a relevance score, an uncertainty estimate, and the frames that support them (§4). Every candidate event, including low-scoring ones, enters the **Semantic Event Buffer** with its clip and assessment.

The **Adaptive Uplink Scheduler** combines these assessments with battery and solar telemetry from the Energy Monitor, the Weather Forecast, and link measurements. It decides which events to upload, keep, or discard, and when to wake the satellite terminal for transmission (§5).

The **Expert-in-the-Loop Workflow** closes the loop (§6). Uploaded clips reach the Cloud Database, where experts review them, and their judgments, frame marks, and written feedback are kept as Expert Annotated Clips and in the Cloud Expert Knowledge. The resulting updates to the expert intent, the knowledge, and the event memory are synchronized back to the edge over the satellite link, while the agent’s weights stay fixed. To keep this feedback representative, ATLAS reserves a small share of each upload budget for sentinel clips, selected by random sampling and by agent uncertainty rather than by score, so that experts also see events the agent rated low.

4 Semantic Assessment Agent

The Semantic Assessment Agent, hereafter the agent, is a fine-tuned language model that assesses each candidate clip against the current expert intent. It reads an assembled textual context and emits a structured assessment that the Adaptive Uplink Scheduler (§5) uses to determine what to upload.

4.1 Context and Assessment

Context assembly. For each candidate event c and expert intent $\mathcal{I}$, the agent reads an assembled context

$$x = (\text{Compile}(\mathcal{I}), \text{Describe}(c), \text{Retrieve}(\mathcal{I}, c), r(c)), \quad (1)$$

where $\text{Retrieve}(\mathcal{I}, c)$ collects the entries retrieved from the expert knowledge store $\mathcal{K}$ and the semantic memory $\mathcal{M}$. The components are as follows.

- **Compiled expert intent.** An offline step compiles the intent $\mathcal{I}$ into a rubric $\text{Compile}(\mathcal{I})$. The rubric has six fixed fields: target, exclude, count, and motion, together with condition, which states when the intent applies, and evidence, which lists the attributes that the descriptions must cover. An open list, criteria, adds free-text checks such as an anomaly descriptor. The rubric is cached and appended to every assessment prompt.
- **Frame descriptions.** $\text{Describe}(c)$ converts every candidate event into text, one line per sampled frame of the form [frame index] [description]. The lightweight DVM, an object detector fine-tuned to the deployment domain with a tracking-by-detection step, runs on every event and contributes classes, counts, positions, and track identities. The heavier VLM adds a holistic description of each frame and runs only when the rubric’s evidence field asks for such context and the DVM reports a class that is not on the rubric’s exclude list.
- **Retrieved expert knowledge.** The expert knowledge store $\mathcal{K}$ is a lightweight, site-specific collection of rules of thumb, seasonal priors, known failure modes (e.g., “bubbles trigger false positives”), and labelling guidelines. The top- k entries most relevant to the intent are retrieved.
- **Retrieved event memory.** The semantic memory $\mathcal{M}$ is a rolling record of the events uploaded from the site, with their frame descriptions, regime labels, and any expert feedback. Records similar to the new event are retrieved, supplying expert corrections and site history.
- **Regime label.** The regime label $r(c)$ names the viewing conditions of the clip, such as clear water, turbid water, low light, or glare. A small on-device classifier assigns it from cheap image statistics such as contrast, blur, and a turbidity proxy, so that the agent knows under which conditions the descriptions were produced.

Assessment interface. Given context x , the agent samples a structured assessment y from its policy $\pi_\theta(y | x)$ with parameters θ ,

$$y = (s, u, \mathcal{F}, \text{rationale}), \quad (2)$$

where s is the relevance score, u is the uncertainty estimate, and $\mathcal{F}$ is the set of frame indices cited as evidence. When multiple assessments of the same event are compared (§4.2), the fields of assessment y are denoted by s_y , u_y , and $\mathcal{F}_y$. The output conforms to a fixed JSON schema with the fields score, uncertainty, evidence_frames, and rationale.

The score $s \in [0, 1]$ is the agent’s estimate of $P(z = 1 | x)$, where $z \in \{0, 1\}$ is the expert’s judgment of whether the clip satisfies the intent. The uncertainty $u \in [0, 1]$ is the agent’s prediction of its own error $|s - z|$. A score near 0.5 means that the evidence in the clip supports either judgment.

A high u means that the agent may be confidently wrong, typically under unfamiliar conditions such as a new site or turbid water, which the regime label $r(c)$ helps it recognize. Every assessment must cite the frames $\mathcal{F}$ whose descriptions support it. The citations tie the score to the observed content of the clip rather than to prior context.

4.2 Training and In-context Adaptation

Cold-start distillation. We first train the agent by knowledge distillation at the sequence level [20] from a stronger teacher model, so that it produces the schema of Eq. (2) from a sensible prior. Each training example pairs a clip with a rubric. Rubrics are either compiled from expert-written intents (§4.1) or synthesized by instantiating the schema with alternative field values, so that distillation covers a broader range of intents than the few written by experts. For each pair, we assemble the context x as at deployment, let the teacher emit a complete assessment, and fine-tune the agent’s backbone to reproduce it. Teacher outputs that contradict Describe(c), for example by citing frames that do not exist, are discarded. This stage alone makes the agent deployable before any expert review.

Refinement with expert judgments. Distillation inherits the teacher’s blind spots, since the teacher has never seen an expert’s verdict on these clips. We therefore refine the agent with expert reviews. To limit expert effort, each review consists of a binary verdict $z \in \{0, 1\}$ on whether the clip satisfies the intent and the frames the expert found decisive for it. A classifier would suffice for the score alone, but the evidence frames and the uncertainty estimate must stay consistent with it and their rewards depend on the decoded output, so we treat the verdict as a reward and refine the single generative policy with Group Relative Policy Optimization (GRPO) [36]. GRPO is the standard optimizer for language-model policies with rule-based rewards [4].

Reward. An assessment y with relevance score s_y , uncertainty estimate u_y , and cited frames $\mathcal{F}_y$ receives the reward

$$\begin{aligned} R(x, y) = & -\text{LogLoss}(s_y, z) \\ & + \rho \text{EvidenceMatch}(y, c) \\ & - \nu(u_y - |s_y - z|)^2, \end{aligned} \quad (3)$$

with weights $\rho, \nu > 0$. LogLoss is the binary cross-entropy, evaluated with s_y clipped to $[\epsilon, 1 - \epsilon]$ for a small constant $\epsilon \in (0, \frac{1}{2})$; as a strictly proper scoring rule, the log-loss term encourages calibrated scores and penalizes overconfident errors. EvidenceMatch(y, c) is the intersection over union of $\mathcal{F}_y$ and $\mathcal{A}(c)$, where $\mathcal{A}(c)$ is the set of frames an expert marked as decisive for z , indexed as in Describe(c), and is defined as one when both sets are empty. The union in the denominator penalizes citing unmarked frames, so when no frames are marked, citing none maximizes the term. The final term is maximized when u_y equals $|s_y - z|$, the realized

absolute error of the relevance score, so the agent learns to report its error as its uncertainty.

Sampling and update. Training inputs are assembled exactly as at deployment. For each training input x , we sample a group of G assessments $\{y_1, \dots, y_G\}$ from the current policy. Because the outputs are sampled, the group differs in score, uncertainty, cited frames, and rationale. Each assessment is scored with the reward of Eq. (3) against the same expert judgment z , and its advantage is the reward normalized within the group. The update scales the log-probability of every token of each sampled assessment by its advantage, raising assessments that scored above the group mean and lowering the rest, subject to a Kullback–Leibler (KL) penalty towards the distilled policy that keeps the refinement close to the prior. The rationale needs no reward term of its own, since rationales that accompany well-rewarded assessments are reinforced through this token-level update.

In-context adaptation. Both training stages run offline on a GPU server before deployment (§7). During deployment, the agent adapts through context updates while its weights remain fixed. Expert feedback is incorporated in two ways. Interpretive feedback is stored for retrieval, either in the event’s record in $\mathcal{M}$ or as a site- or season-specific heuristic in $\mathcal{K}$. Visual cues, such as markings that distinguish two species, are added to the rubric’s evidence field. The rubric is then recompiled so that the VLM includes these cues in subsequent descriptions. Changes in expert intent also trigger rubric recompilation. The updated rubric and retrieval stores are synchronized to the edge via satellite.

5 Adaptive Uplink Scheduler

The Adaptive Uplink Scheduler decides which buffered events to upload and when, under intermittent connectivity and limited energy. We formulate upload control as a Constrained Markov Decision Process (CMDP) [1] that balances delivered semantic value, data freshness, and battery reserves. The scheduler combines the agent’s scores and uncertainty estimates with buffer state, energy telemetry, and forecasts. A runtime safety shield checks each resulting upload plan against projected energy availability and connectivity.

5.1 State, Action, and Transition

State. Decisions are made at discrete steps t , one per decision interval $[T_t, T_{t+1})$, where T_t is the clock time at the start of step t . The buffer, battery telemetry, and forecasts indexed by t are evaluated at T_t , before the action of step t is applied, whereas the plan and upload outcomes of step t refer to the interval that follows. Let $\mathcal{B}_t$ be the set of pending events in the Semantic Event Buffer at step t . Each pending event $i \in \mathcal{B}_t$ carries its capture time T_i^{cap} and the score s_i and uncertainty estimate u_i that the agent assigned when the event entered the buffer (§4.1), which are not revised while it is pending. At step t , the scheduler’s policy observes

a compact summary $\mathbf{o}_t$ of this state:

$$\mathbf{o}_t = \left(\underbrace{\text{SOC}_t, \widehat{H}_{t:t+h-1}}_{\text{energy state and forecast}}, \underbrace{\widehat{\text{BW}}_{t:t+h-1}, q_t, \bar{L}_t, \ell_t^{\max}}_{\text{link forecast and buffer state}}, \right. \\ \left. \underbrace{p_t^{\text{prio}}, \bar{u}_t, \bar{s}_t}_{\text{semantic context}}, \underbrace{\tau_t, \text{DoY}_t}_{\text{time context}} \right). \quad (4)$$

Here $\text{SOC}_t \in [0, 1]$ is the battery state of charge, and $\widehat{H}_{t:t+h-1}$ and $\widehat{\text{BW}}_{t:t+h-1}$ are the predicted energy harvest and uplink bandwidth for each of the next h decision intervals. The pending payload volume is q_t bytes. The data lag $\bar{L}_t$ is the mean age $T_t - T_t^{\text{cap}}$ of the pending events, and $\ell_t^{\max}$ is their maximum age. The quantity p_t^{prio} is the fraction of pending events whose score exceeds a fixed reference level. Time of day τ_t and day of year DoY_t provide temporal context. The mean score $\bar{s}_t$ and mean uncertainty $\bar{u}_t$ over a nonempty buffer are

$$\bar{s}_t = \frac{1}{|\mathcal{B}_t|} \sum_{i \in \mathcal{B}_t} s_i, \quad \bar{u}_t = \frac{1}{|\mathcal{B}_t|} \sum_{i \in \mathcal{B}_t} u_i, \quad (5)$$

so that $\bar{s}_t$ and p_t^{prio} together convey the shape of the score distribution. For an empty buffer, $\bar{s}_t, \bar{u}_t, p_t^{\text{prio}}, \bar{L}_t$, and $\ell_t^{\max}$ are set to zero, and $q_t = 0$ distinguishes this case from a buffer whose events all scored zero. The policy sees only this summary, whereas plan construction reads the per-event scores, sizes, and sentinel marks in $\mathcal{B}_t$ directly.

Energy reference. The scheduler estimates the energy stored in the battery at the start of step t , and the reserve, as

$$E_t = \text{SOC}_t E^{\text{cap}}, \quad E_{\min} = \text{SOC}_{\min} E^{\text{cap}}, \quad (6)$$

where E^{cap} is the usable battery capacity and $\text{SOC}_{\min} \in (0, 1)$ is the reserve state of charge. The software reserve $E_{\min}$ is set above the battery management system's protective low-voltage cutoff and is the common reference for the reward, the constraint cost, and the shield's energy guard. Only SOC_t enters $\mathbf{o}_t$, since E_t is a fixed multiple of it.

Action. The control policy π_ϕ , with parameters ϕ , outputs $a_t = [b_t, \gamma_t]$, where $b_t \geq 0$ is the total upload budget in bytes and $\gamma_t \in [0, 1]$ is the score threshold for retaining an event and making it eligible for upload, distinct from the fixed reference level used for p_t^{prio} . Uploads use a representation κ from a fixed ladder ordered by payload size: a keyframe set, a reduced-resolution clip, or the full clip. Rather than being learned, κ is set at each rubric compilation to the smallest representation that lets experts verify the rubric (§4.1), such as a keyframe set when the rubric constrains only appearance and a clip when its motion or count field requires experts to see movement. The choice of κ therefore affects which events fit within the budget but not the value credited to an individual event in Eq. (9). Let w_i denote the payload size of event i in bytes under the current κ , recomputed for every pending event before the next decision step whenever

a recompilation changes κ , so that $q_t = \sum_{i \in \mathcal{B}_t} w_i$. Ordinary events with $s_i < \gamma_t$ are discarded from the buffer, and those with $s_i \geq \gamma_t$ are considered for transmission. Sentinel events, which are designated for expert audit by random sampling and by agent uncertainty (§6) before discarding and keep that designation while pending, are exempt from the discard rule and are served first through the audit allowance below.

The audit allowance b_t^{sen} is a fixed fraction of b_t rather than a policy output. Serving pending sentinel events in order of capture time, the scheduler selects a set $\mathcal{P}_t^{\text{sen}}$ whose planned payload d_t^{sen} fits within the allowance:

$$d_t^{\text{sen}} = \sum_{i \in \mathcal{P}_t^{\text{sen}}} w_i \leq b_t^{\text{sen}}. \quad (7)$$

Sentinel events that do not fit can still enter the task-directed plan if they pass the threshold, and any sentinel event not delivered remains pending. The remaining budget goes to task-directed uploads:

$$\mathcal{P}_t^{\text{task}} = \text{SelectTop}(b_t - d_t^{\text{sen}}, \gamma_t; \mathcal{B}_t \setminus \mathcal{P}_t^{\text{sen}}). \quad (8)$$

SelectTop keeps the candidates with $s_i \geq \gamma_t$ and ranks them by the confidence-weighted score $s_i(1 - u_i)$, so that among events with similar scores, those assessed more confidently rank higher. Uncertain events still reach experts through the audit allowance, which targets high uncertainty (§6). Breaking ties in favour of older events, SelectTop traverses the ranked list and adds each event whose size fits within the remaining budget, skipping those that do not, so that one large event cannot leave the budget unused.

The upload plan $\mathcal{P}_t = \mathcal{P}_t^{\text{sen}} \cup \mathcal{P}_t^{\text{task}}$ therefore contains each event at most once and has total payload $\sum_{i \in \mathcal{P}_t} w_i \leq b_t$. Before transmission, the safety shield (§5.4) maps it to the executable plan $\mathcal{P}_t^{\text{safe}} = \text{Shield}(\mathcal{P}_t, \mathbf{o}_t)$, which is either $\mathcal{P}_t$ itself or the empty plan. Ordinary events that pass the threshold but are not successfully uploaded remain pending and are compared against the threshold of the next step.

Transition. After the policy proposes a_t , the scheduler applies the discard rule to $\mathcal{B}_t$, constructs $\mathcal{P}_t$, and executes $\mathcal{P}_t^{\text{safe}}$. The next state, with buffer $\mathcal{B}_{t+1}$ and energy E_{t+1} , reflects discarded events, completed uploads, new arrivals, realized harvest, and device load, with energy accounting covering both task-directed and sentinel traffic and the associated radio-state transitions. Delivered and discarded events leave the buffer. Retained events keep their capture times and continue to age, including through a shutdown, whereas observations missed during a shutdown are lost coverage and do not enter $\bar{L}_t$.

Satellite power-state control. ATLAS switches the terminal between sleep and wake through Starlink's local gRPC interface, according to the executable plan and connectivity checks. Our measurements of the Starlink Standard (Gen 3) UT show power draws of 26.3 W in sleep, 49.3 W while idle, and 66.4 W during active transmission, with a peak of

Algorithm 1 Constrained SAC for ATLAS Upload Control

```
1: Initialize actor  $\pi_\phi$  and twin critics  $Q_{\psi_1}, Q_{\psi_2}$ 
2: Initialize target critics,  $\lambda \geq 0$ , and replay buffer  $\mathcal{D}$ 
3: for each multi-day training trace do
4:   Initialize battery and event-buffer state
5:   for each decision step  $t$  do
6:     Upload control
7:     Observe  $\mathbf{o}_t$ ; sample  $a_t \sim \pi_\phi(\cdot | \mathbf{o}_t)$ 
8:     Update sentinel designations in  $\mathcal{B}_t$ 
9:     Discard non-sentinel events with  $s_i < \gamma_t$ 
10:    Build  $\mathcal{P}_t = \mathcal{P}_t^{\text{sen}} \cup \mathcal{P}_t^{\text{task}}$  by Eqs. (7) and (8)
11:     $\mathcal{P}_t^{\text{safe}} \leftarrow \text{Shield}(\mathcal{P}_t, \mathbf{o}_t)$ 
12:    Execute  $\mathcal{P}_t^{\text{safe}}$  and observe  $\mathbf{o}_{t+1}$ 
13:    Compute  $J_t, \mathbf{C}_t$ , and terminal indicator  $m_t$ 
14:    Append the transition to  $\mathcal{D}$ 
15:    Policy learning
16:    Sample a minibatch from  $\mathcal{D}$ 
17:    Compute minibatch mean cost  $\bar{\mathbf{C}}$ 
18:    Compute penalized rewards by Eq. (13)
19:    Update twin critics on the penalized rewards
20:    Update  $\pi_\phi$  using the soft-critic objective
21:     $\lambda \leftarrow [\lambda + \eta_\lambda (\bar{\mathbf{C}} - \delta)]_+$ 
22:    Soft-update target critics
23:   end for
24: end for
```

127.6 W, and waking to a usable link takes 183 s at 50.6 W, or approximately 9.26 kJ. Because the terminal returns to sleep at the end of each decision interval, every nonempty executed plan incurs this wake-up cost, which the scheduler accounts for with the idle and active energy that follows.

5.2 Reward and Constraints

Delivered semantic value. Let $\mathcal{U}_t \subseteq \mathcal{P}_t^{\text{safe}}$ be the set of events successfully uploaded during step t . Their delivered semantic value is

$$V_t = \sum_{i \in \mathcal{U}_t} s_i (1 - u_i), \quad (9)$$

the confidence-weighted score that SelectTop ranks by. Since s_i estimates the probability that an expert would judge event i to satisfy the intent and u_i predicts the error of that estimate (§4.1), V_t is a confidence-weighted count of the delivered clips expected to satisfy the intent.

Reward. The per-step reward J_t credits delivered value, charges data lag, and credits battery energy above the reserve while charging any shortfall below it:

$$J_t = \omega_V V_t - \omega_L \bar{L}_t - \omega_E (E_{\min} - E_t), \quad (10)$$

with nonnegative weights ω_V , ω_L , and ω_E . Here V_t reflects the executed plan, whereas $\bar{L}_t$ and E_t describe the start of step t , so the effect of an action on lag and energy appears in the reward and costs of later steps.

Constraints. Two constraint costs measure excess lag and energy-reserve shortfall:

$$\begin{aligned} C_t^{(1)} &= \max(0, \bar{L}_t - L^{\text{tgt}}), \\ C_t^{(2)} &= \max(0, E_{\min} - E_t), \end{aligned} \quad (11)$$

where L^{tgt} is the data-lag target. With $N \geq 1$ denoting the number of decision steps in an episode, the scheduler targets the constrained objective

$$\begin{aligned} \max_{\pi_\phi} \mathbb{E}_{\pi_\phi} \left[\sum_{t=0}^{N-1} \beta^t J_t \right] \\ \text{s.t. } \frac{\mathbb{E}_{\pi_\phi} \left[\sum_{t=0}^{N-1} C_t^{(j)} \right]}{\mathbb{E}_{\pi_\phi} [N]} \leq \delta_j, \quad j \in \{1, 2\}, \end{aligned} \quad (12)$$

where $\beta \in [0, 1)$ is the discount factor, $\delta_j \geq 0$ is the budget on the mean per-step cost of constraint j , and expectations are taken over episode trajectories under π_ϕ . Each constraint weights every visited decision step equally across episodes and reduces to the expected per-episode mean cost when all episodes have the same length.

5.3 Policy Learning

We train the scheduler with a Lagrangian-style Soft Actor-Critic (SAC) procedure [10, 11, 55], following reward constrained policy optimization [41]. The constraint costs of Eq. (11) are folded into a penalized reward

$$\tilde{J}_t = J_t - \lambda^\top \mathbf{C}_t, \quad \mathbf{C}_t = [C_t^{(1)}, C_t^{(2)}]^\top, \quad (13)$$

where $\lambda = (\lambda_1, \lambda_2)^\top \geq 0$ are the Lagrange multipliers. SAC trains the actor π_ϕ and twin critics Q_{ψ_1} and Q_{ψ_2} , with parameters ψ_1 and ψ_2 , to maximize the entropy-regularized discounted return of $\tilde{J}_t$, so the penalty acts on discounted costs and steers the policy toward the mean-cost constraints of Eq. (12) rather than encoding them exactly. The multipliers are updated by projected dual ascent [41] at rate $\eta_\lambda > 0$, rising whenever the minibatch mean constraint cost $\bar{\mathbf{C}}$, an off-policy proxy for the mean per-step cost in Eq. (12), exceeds its budget $\delta = (\delta_1, \delta_2)^\top$. Algorithm 1 summarizes the procedure, where $[\cdot]_+$ replaces negative components with zero and leaves nonnegative components unchanged. Three aspects are specific to ATLAS.

Continuous action. SAC optimizes both outputs of π_ϕ directly, and the events to upload follow deterministically from them through plan construction (§5.1).

Learning under the safety shield. Each record in the replay buffer $\mathcal{D}$ stores the observation $\mathbf{o}_t$, the proposed action a_t , the reward J_t , the costs $\mathbf{C}_t$, the next observation $\mathbf{o}_{t+1}$, and an episode-termination indicator $m_t \in \{0, 1\}$, equal to one if the episode ends after step t . The record keeps the proposed action rather than the executed plan $\mathcal{P}_t^{\text{safe}}$, while V_t and $\mathbf{o}_{t+1}$ reflect score-based discarding and any intervention by the shield (§5.4), so the policy learns the consequences of its

proposals as they are actually executed. Storing J_t and C_t separately lets every update apply the current multipliers.

Training data and deployment. Training uses multi-day trace-driven episodes built from deployment records of solar harvest, device load, uplink throughput and outages, and event arrivals. During replay, energy and buffer states evolve under the executed upload plans and the device-load accounting, exposing the policy to extended periods of resource scarcity. We initialize training with experience from historical operation and baseline-derived threshold policies, then optimize the scheduler off-policy. During deployment, π_ϕ is refined with conservative exploration under the same shield, while the agent’s weights stay fixed (§4.2). This refinement is separate from the expert-feedback workflow, which adapts semantic assessment through the rubric and the retrieval stores.

5.4 Runtime Safety Shield

The constraints of Eq. (12) are imposed only in expectation, and the policy keeps exploring after deployment, so ATLAS also enforces per-step limits that every executed plan must satisfy under the current forecasts. Before execution, the shield checks two guards and defers the plan if either fails:

1. **Energy protection.** The projected battery energy at the end of the interval, $\widehat{E}_{t+1}^{\text{post}}$, must not fall below $E_{\min}$. The projection combines E_t , the harvest forecast, and the estimated terminal and device energy for the plan, including the wake-up cost.
2. **Connectivity feasibility.** The link must not be predicted to be unavailable, such as under severe weather conditions [9].

The shield therefore yields $\mathcal{P}_t^{\text{safe}} \in \{\mathcal{P}_t, \emptyset\}$ for task-directed and sentinel traffic alike, and the check is repeated just before transmission begins. A deferred plan leaves the terminal asleep, and its events remain pending. Because the guards act on forecasts, a harvest shortfall can still drain the battery (§9). Data lag is not enforced by the shield. It is addressed during learning through the mean-cost constraint on $C_t^{(1)}$ in Eq. (12) and is influenced at execution time by the age tie-break of SelectTop.

6 Expert-in-the-Loop Workflow

Human experts review uploaded clips in the cloud, and their feedback returns to the edge as updates to the context the agent reads (§4.2). Such review is standard practice in fisheries monitoring: trained reviewers annotate and audit electronic-monitoring video [28], and automatic salmon counters are validated by manual comparison with video [26]. ATLAS builds on this existing review rather than adding a new one, and turns each review into feedback for the agent.

Sentinel sampling. Task-directed uploads include only clips that pass the score threshold, so reviewing them alone

is unlikely to reveal relevant events assigned low scores. ATLAS therefore selects sentinel candidates independently of their relevance scores through two complementary mechanisms. Random candidates are sampled within each regime defined by $r(c)$ to provide review coverage across the score range and observed conditions. Uncertainty-flagged candidates have high u , indicating that the agent expects their scores to be less reliable. Both types are eligible for the audit allowance described in §5.1 regardless of score.

Expert review. Experts review both task-directed and sentinel uploads on the cloud platform (§7). For each clip, an expert records the judgment z of whether it satisfies the current intent and marks the frames $\mathcal{A}(c)$ that decide it (§4.2). Experts can also add short comments on discriminative cues or failure causes, such as “tail spots visible” or “glare/occlusion.”

Recurring feedback. Feedback from our collaborating partner falls into four categories.

- **Fine-grained visual discriminators.** Experts annotate cues such as tail and back markings, gumline colour, and body coloration. These cues extend the rubric’s evidence field, so that the VLM reports them in every subsequent frame description.
- **Counting under occlusion and bursty passage.** Experts explain how to count when tracks overlap or break during dense passage, and when a re-entering track belongs to an individual already counted. These rules are added to $\mathcal{K}$ and applied to the counts and track identities in Describe(c).
- **Contextual false-positive signatures.** Comments such as “debris drift,” “air bubbles,” “glare band,” and “turbidity plume” are stored with the rejected clips as negative exemplars in $\mathcal{M}$, and recurring ones are added to $\mathcal{K}$ as known failure modes, so that similar observations receive lower scores.
- **Intent-based relevance.** Guidance on first arrivals, rare species, and unusual behaviour helps the agent judge whether an observation adds evidence for the current intent or repeats a routine event. Such guidance is added to $\mathcal{K}$ as seasonal priors and rules of thumb, and, when it refines the intent itself, enters the rubric’s criteria list at recompilation.

Synchronization. Expert updates are batched in the cloud and delivered to the edge during the next upload window, so that synchronization never wakes the terminal on its own. A recompiled rubric is the only update that affects uploads directly, since the representation κ is chosen from it (§5.1). All other updates act only indirectly, by changing the scores s and uncertainty estimates u that the agent assigns to subsequent events.

7 Evaluation Methodology

Deployment and data. We deployed ATLAS at 18 off-grid salmon monitoring sites operated with our fisheries partner (Figure 8) from April to November 2025, spanning the full salmon migration and spawning season. Each site ran the hardware stack shown in Figure 1d. Throughout the deployment, every site recorded solar irradiance, weather, device load, uplink bandwidth, and camera footage containing both events and non-events. These recordings, together with the clips the partner had annotated before the deployment and its written domain knowledge, are the data behind every result in this paper. ATLAS ran live at the sites throughout, with experts reviewing uploads on the partner’s platform hosted on an Amazon Web Services server. Each site has a single terminal and battery, and its conditions never repeat, so competing systems cannot be compared in the field. All head-to-head comparisons therefore replay the recorded traces on the deployment hardware, so that every system faces identical conditions.

Trace-driven replay. Replay runs on an NVIDIA Jetson Orin NX [29] with 16 GB of shared CPU-GPU memory and 1 TB of local storage for video buffering, the same device as at the sites. ATLAS runs the entire on-device pipeline of Figure 7 on the Jetson. The recorded footage supplies the candidate events, and the recorded irradiance, weather, and bandwidth traces drive the simulated battery and link. Because the tasks require different event types and temporal contexts, no single recording window suits all of them. For each site and task, we therefore sample replay traces from the site’s recordings suited to that task and fix them, so that ATLAS and every baseline run on the same traces, battery capacity, initial state of charge, and evaluation window, while each system’s own execution determines its energy consumption, battery trajectory, and uptime. Replay traces are disjoint in time from the traces used to train ATLAS’s scheduler and the Joint-optimized policy and from the footage used to fine-tune the DVM.

Implementation. The DVM is a YOLO26s detector [43] fine-tuned on annotated footage from the deployment sites, with ByteTrack [65] for tracking, and the VLM is FastVLM-1.5B [45]. The agent’s backbone is Qwen3-8B [54] in non-thinking mode, fine-tuned with LoRA [13] of rank 32 and quantized to 4-bit weights with AWQ [23], so that it fits in the Jetson’s shared memory alongside the DVM and the VLM [42]. Qwen3-Embedding-0.6B [64] embeds the entries of $\mathcal{K}$ and $\mathcal{M}$ and the retrieval queries. Distillation used GPT-4o [30] as the teacher, and refinement used 1,000 expert-reviewed clips disjoint from the evaluation traces, with $G = 8$ samples per input at temperature 1.0, $\rho = 0.5$, $\nu = 1$, $\epsilon = 10^{-3}$, and a KL coefficient of 0.03. Both stages ran on a server with four NVIDIA RTX A5000 GPUs.

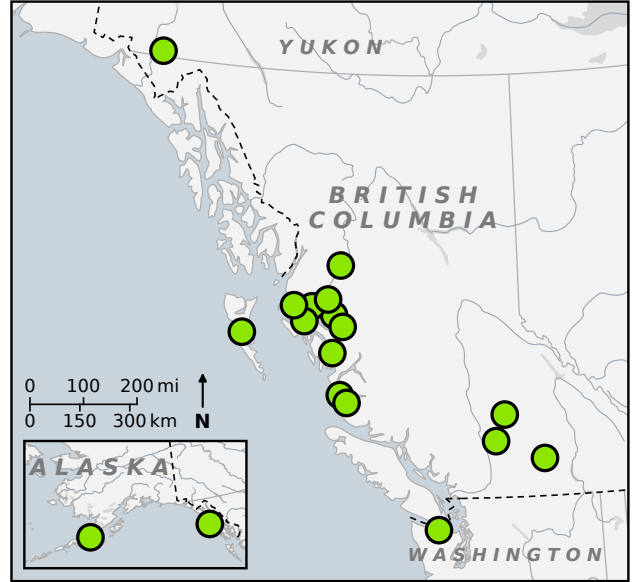


Figure 8. Geographic overview of ATLAS deployment sites operated in collaboration with our salmon fisheries partner across North America.

Baselines. We compare against two content-selection strategies that existing systems rely on, motion triggering and on-device vision, and against a baseline that adds learned energy-aware scheduling. The first two baselines upload the selected clips whenever the battery state of charge exceeds the reserve $\text{SOC}_{\min}$ (§5.1), the same reserve that ATLAS’s safety shield protects, and the third replaces this rule with a learned policy. For each task, the experts specify the target classes and tune per-site confidence thresholds for the detector-based baselines, the closest approximation of the task that these systems can take as input. The ablation in §9 separates the contribution of each ATLAS component.

- **Motion-based.** Motivated by on-camera filtering [22], this baseline selects clips by motion, using background subtraction with a mixture-of-Gaussians model.
- **Vision-based.** Motivated by InFi [58], this baseline selects clips with the same DVM as ATLAS, by the expert-specified classes and thresholds.
- **Joint-optimized.** Inspired by learned video-quality adaptation [25, 40] and energy-aware transmission scheduling [35], this baseline pairs the same DVM with an RL policy trained offline on the same deployment traces as ATLAS’s scheduler. The policy takes the detection confidence as the relevance score, together with the battery state of charge, available bandwidth, and queue state, and decides whether to upload each clip and at what quality.

Expert tasks. To evaluate ATLAS under realistic expert-driven workflows, we worked with our partners to define four expert tasks drawn from real-world salmon monitoring deployments, as detailed below.

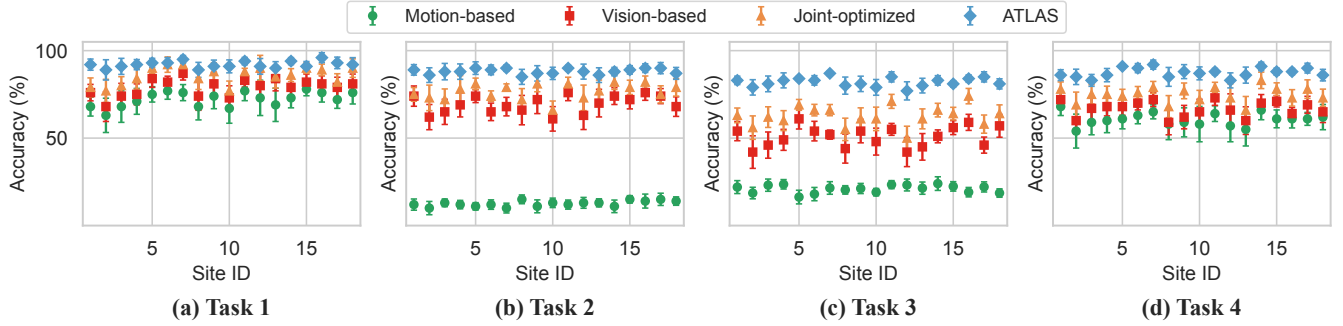


Figure 9. Clip-selection accuracy across sites for different tasks.

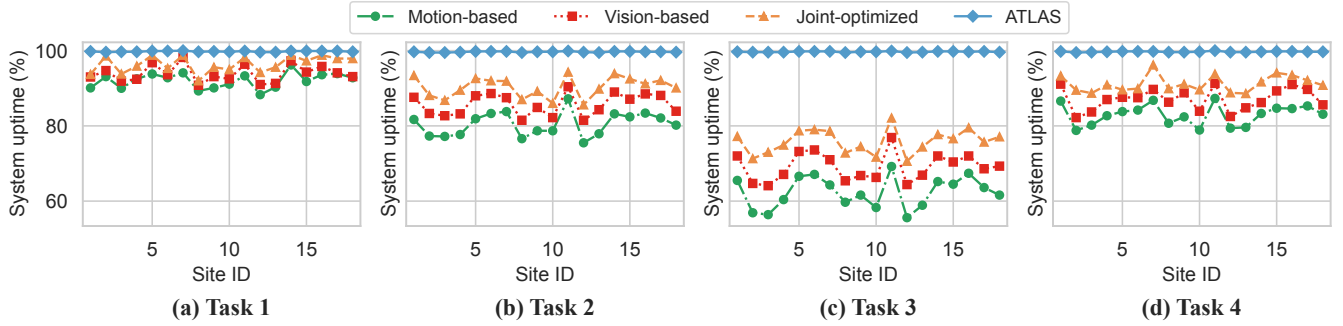


Figure 10. System uptime across sites for different tasks.

Table 1. ATLAS versus three baselines across 18 deployment sites under four expert tasks. Accuracy is the percentage of candidate clips whose upload decision agrees with the expert; uptime is the percentage of the evaluation window in which the device is not shut down by the battery cutoff.

Metric / System	Task 1	Task 2	Task 3	Task 4
Accuracy (%)				
Motion-based	72.3	13.0	21.0	60.7
Vision-based	79.1	69.4	50.8	66.7
Joint-optimized	85.7	76.6	62.6	74.8
ATLAS	92.1	88.2	82.0	87.4
Uptime (%)				
Motion-based	92.1	80.5	62.4	82.9
Vision-based	93.9	85.7	69.1	87.1
Joint-optimized	96.5	90.3	75.9	91.2
ATLAS	99.9	99.7	99.9	99.7

Task 1: Target Species Detection. Identify and upload video clips that contain a specific target salmon species (e.g., Chinook salmon).

Task 2: Non-Target and Obstruction Identification. Identify and upload clips containing non-target species or obstructions that may interfere with salmon passage observations.

Task 3: Passage Activity Monitoring. Identify and upload video clips that help experts assess changes in salmon passage activity throughout the day, including increases, periods of high salmon passage activity, and decreases.

Task 4: Directional Migration Filtering. Upload video clips that capture upstream salmon passage while filtering downstream movement and non-migratory behaviour.

Metrics. Clip-selection accuracy is the percentage of candidate clips whose outcome, uploaded within the evaluation window or not, agrees with the expert’s judgment z , so it penalizes both irrelevant uploads and missed relevant clips. Every candidate clip in the replay set carries such a judgment, collected independently of the clips used for refinement (§4.2), and sentinel uploads are counted like any other. *System uptime* is the percentage of the evaluation window in which the edge device is not shut down by the battery cutoff. Both are reported per site and task. *Data lag*, the mean age $\bar{L}_t$ of pending events (§5.1), is shown over time in §8.3.

8 Performance Evaluation

8.1 Overall Performance

Table 1 summarizes the overall performance of ATLAS compared with three representative baselines across 18 real-world deployments and four expert-defined tasks. We evaluate performance along two key dimensions that are important for long-term remote monitoring: system uptime and clip-selection accuracy.

System uptime. ATLAS maintains near-continuous operation, with task-level uptime ranging from 99.7% to 99.9% across the four evaluated workloads. In contrast, baseline systems exhibit substantial downtime under energy-constrained conditions, with motion-based approaches degrading to as low as 62.4% uptime in Task 3. Even the joint-optimized baseline, which explicitly accounts for system constraints, experiences non-trivial outages (up to 24.1%). These results indicate that static or offline-optimized policies are insufficient to handle prolonged low-energy periods and task-dependent

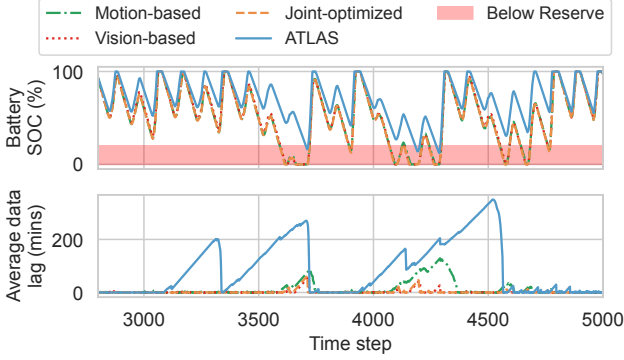


Figure 11. Battery SOC (top) and average data lag (bottom) of ATLAS and the baselines during a prolonged rainy period.

workload variation, whereas ATLAS proactively adapts its uplink behaviour to both its energy state and expert intent.

Clip-selection accuracy. ATLAS achieves the highest accuracy on all four tasks, exceeding the strongest baseline on each by 6.4 to 19.4 percentage points. The margin is largest on Task 3, where ATLAS reaches 82.0%, compared with 62.6% for Joint-optimized and 21.0% for Motion-based. Averaged equally across tasks, ATLAS achieves a mean accuracy of 87.4%, 26.4 percentage points above the baseline mean. Together with the uptime results, these findings show that ATLAS improves accuracy while maintaining near-continuous operation under the evaluated energy constraints.

8.2 Performance Across Sites and Tasks

Accuracy. Figure 9 shows the per-site accuracy behind the averages in Table 1, from the same replay traces (§7). ATLAS achieves the highest accuracy at every site and on every task, with low variance across sites. Its margin over the baselines is smallest on Task 1 but consistent across sites. The advantage is largest on Task 3, which requires clips that help experts assess changes in passage activity throughout the day: baseline accuracy drops sharply and varies widely across sites, while ATLAS remains stable, showing robustness to complex temporal and contextual requirements.

System uptime. Figure 10 presents the site-level uptime results for the four task-specific replay workloads. The corresponding task-level aggregates in Table 1 range from 99.7% to 99.9% for ATLAS. These results demonstrate near-continuous operation under the evaluated conditions, alongside the improvements in clip-selection accuracy.

Summary. Across heterogeneous sites and increasingly complex tasks, ATLAS delivers consistently high accuracy while maintaining near-continuous operation.

8.3 Performance During Worst-Case Seasonal Conditions

Figure 11 illustrates system behaviour during one of the most challenging seasonal periods, characterized by prolonged rainfall and limited solar input. Under these conditions, battery SOC falls repeatedly below the reserve and down to

Table 2. Ablation results for ATLAS.

Metric / Variant	Task 1	Task 2	Task 3	Task 4
Accuracy (%)				
w/o upload scheduler	90.0	86.0	79.0	85.1
w/o semantic assessment agent	85.0	77.1	68.2	76.5
w/o expert-in-the-loop	89.9	82.7	78.9	83.5
ATLAS	92.1	88.2	82.0	87.4
Uptime (%)				
w/o upload scheduler	93.2	92.1	87.0	94.1
w/o semantic assessment agent	99.2	98.4	96.8	98.9
w/o expert-in-the-loop	94.1	93.2	95.4	97.5
ATLAS	99.9	99.7	99.9	99.7

the cutoff across all baselines, causing frequent system shut-downs. In contrast, ATLAS maintains battery SOC above the reserve throughout the period by proactively budgeting energy and deferring video uploads. This energy stability comes at the cost of increasing data lag as deferred uploads accumulate during sustained energy scarcity. Once energy availability recovers, ATLAS clears the accumulated upload backlog, returning data lag to near zero while preserving uninterrupted system operation.

9 Ablation Study

We evaluate three ablation variants, each removing one ATLAS component. *W/o upload scheduler* keeps the semantic assessment agent and expert updates but replaces the scheduler with a first-come, first-served policy: clips that pass a fixed score threshold are uploaded in arrival order whenever the battery state of charge is above the safety shield’s reserve threshold. *W/o semantic assessment agent* keeps the upload scheduler, safety shield, and expert updates, but scores each clip by the DVM’s confidence for the target classes and sets its uncertainty estimate to zero. Since this variant has no agent context to update, expert feedback instead tunes the DVM’s target classes and confidence thresholds. *W/o expert-in-the-loop* removes all expert updates while preserving the rest of the ATLAS runtime.

Table 2 reports accuracy and uptime on the four tasks. The full system is the most accurate on every task, with a mean accuracy of 87.4%. Removing the semantic assessment agent causes the largest loss, lowering mean accuracy to 76.7%, a drop of 10.7 percentage points that ranges from 7.1 to 13.8 points across tasks. The loss is largest on Task 3, whose temporal and contextual requirements cannot be expressed through DVM classes and thresholds, even with expert tuning. Removing expert-in-the-loop updates lowers mean accuracy to 83.8%, a drop of 3.7 points, and removing the upload scheduler lowers it to 85.0%, a drop of 2.4 points.

Removing the upload scheduler causes the largest uptime loss. Mean uptime falls from 99.8% to 91.6%, a decrease of 8.2 points, with the largest drop on Task 3. Without the

scheduler’s forward-looking energy budget, uploads continue whenever the battery is above the reserve threshold, so the battery is often close to that threshold when a low-harvest period begins.

Removing expert-in-the-loop updates reduces uptime by 2.2 to 6.5 points across tasks, with a mean decrease of 4.8 points, even with the scheduler and safety shield enabled. Without updated expert knowledge, irrelevant clips can receive high relevance scores, and uploading them leaves less energy for low-harvest periods. Because the energy guards rely on harvest predictions, a harvest shortfall can still exhaust the battery after uploads are suspended, so earlier upload decisions affect uptime even with the guards in place.

In contrast, removing the semantic assessment agent reduces uptime by only 0.7 to 3.1 points, with a mean decrease of 1.5 points. Expert-tuned DVM thresholds make this variant conservative: its accuracy loss comes mainly from relevant clips it fails to select, not from irrelevant clips it uploads, so it spends less energy on uploads.

Together, these results separate the roles of the three components. The semantic assessment agent contributes most to accuracy, the upload scheduler contributes most to uptime, and expert-in-the-loop updates improve both. Expert tuning of DVM thresholds does not recover the agent’s accuracy, and no variant matches the full system on either metric.

10 Related Work

Energy- and connectivity-constrained sensing. Intermittent backhaul and reliance on harvested energy are long-standing constraints on unattended sensor deployments. Delay-tolerant architectures move data opportunistically rather than assuming a continuous end-to-end path [5, 7, 17, 31, 52], operating systems manage energy as an explicit, allocatable resource [59], energy-neutral systems pair harvesting with multi-stage storage for perpetual operation [16] and adapt duty cycles and application activity to predicted harvest [18, 39], and transmission schedulers defer sending to moments when it is cheaper or more informative [35, 51]. In wildlife monitoring, edge processing with cloud support shrinks the data that must cross constrained links [6], and a recent proposal keeps expert knowledge in an explicit knowledge base on the edge so that adaptation does not require retraining [21]. None of these systems weighs what a transmission is worth to the experts who use the data beyond a fixed proxy, a priority level, an object class, or a measure of information. ATLAS determines what each clip is worth at runtime from expert intent that changes over a deployment.

Edge/on-camera video analytics and bandwidth reduction. Systems research has developed rich mechanisms to reduce video analytics cost by dynamically trading accuracy for resource usage through configuration adaptation and runtime control [12, 15, 60, 61], and by filtering

frames early, often close to the camera, to avoid processing or transmitting uninformative content [6, 12, 14, 22, 38]. Zero-streaming cameras go further, storing video locally and transferring it only when an analyst issues a query [53]. These approaches typically optimize generic accuracy and throughput objectives or respond to explicit queries, whereas ATLAS decides proactively what to upload based on *expert intent* and co-optimizes this decision against energy-driven uptime constraints.

LEO satellite networking. Measurement, simulation, and emulation studies have characterized LEO network behaviour, from constellation-scale dynamics [9, 19] and the energy, thermal, and network conditions of satellite computing [47] to the performance of Starlink’s network and applications [8, 27]. Other work advances satellite communication architectures. L2D2 uses distributed ground stations to reduce data-delivery latency from Earth-observation satellites [44]. SateRIoT combines link estimation and inter-gateway data sharing to improve the energy efficiency and performance of satellite IoT backhaul [34], while MOSAIC supports multi-tenant direct-to-cell satellite access [24]. At the application layer, StarStream adapts live video analytics to Starlink uplink variability through Transformer-based throughput prediction and content-aware configuration optimization [63]. ATLAS addresses the complementary problem of operating the terminal under a variable solar-energy budget, accounting for its wake-up and operating energy so that uploads are planned only when the site can afford them.

11 Conclusion

We presented ATLAS, an expert-guided, energy-aware system that treats satellite backhaul as a scarce resource, combining FM-based interpretation of expert intent in natural language with constrained upload scheduling. In trace-driven comparisons on salmon monitoring data covering an entire migration and spawning season at 18 off-grid river sites, ATLAS achieved 87.4% mean clip-selection accuracy across four expert-defined tasks, 26.4 percentage points above the baseline mean, with the largest gains on complex tasks. It also maintained near-continuous operation through low-solar periods, whereas baselines averaged 16.0% downtime. More broadly, these results suggest that robust remote wildlife monitoring requires joint design of domain-grounded semantic reasoning and resource-aware control, not efficient edge inference or opportunistic connectivity alone.

Acknowledgments

We thank the anonymous reviewers and our shepherd for their constructive feedback and guidance on this paper. We are also grateful to the Pacific Salmon Foundation and the Wild Salmon Center for their support and collaboration. This research was supported by an NSERC Discovery Grant and a Mitacs Accelerate Grant.

References

- [1] Eitan Altman. 2021. *Constrained Markov decision processes*. Routledge.
- [2] Sara Beery, Grant Van Horn, and Pietro Perona. 2018. Recognition in terra incognita. In *Proc. ECCV*. 456–473.
- [3] Raúl de La Fuente and Thomas Watteyne. 2023. Hands-on Evaluation of Kinéis Satellite IoT Technology. In *Proc. ACM IPSN*. 334–335.
- [4] DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint arXiv:2501.12948* (2025).
- [5] Vladimir Dyo, Stephen A Ellwood, David W Macdonald, Andrew Markham, Cecilia Mascolo, Bence Pásztor, Salvatore Scellato, Niki Trigoni, Ricklef Wohlers, and Kharsim Yousef. 2010. Evolution and sustainability of a wildlife monitoring sensor network. In *Proc. ACM SenSys*. 127–140.
- [6] Andy Rosales Elias, Nevena Golubovic, Chandra Krintz, and Rich Wolski. 2017. Where’s the bear? Automating wildlife image processing using IoT and edge cloud systems. In *Proc. IEEE IoTDI*. 247–258.
- [7] Kevin Fall. 2003. A delay-tolerant network architecture for challenged internets. In *Proc. ACM SIGCOMM*. 27–34.
- [8] Hao Fang, Haoyuan Zhao, Jianxin Shi, Miao Zhang, Guanzhen Wu, Yi Ching Chou, Feng Wang, and Jiangchuan Liu. 2024. Robust Live Streaming over LEO Satellite Constellations: Measurement, Analysis, and Handover-Aware Adaptation. In *Proc. ACM MM*. 5958–5966.
- [9] Hao Fang, Haoyuan Zhao, Feng Wang, Yi Ching Chou, Long Chen, Jianxin Shi, and Jiangchuan Liu. 2025. Streaming media over LEO satellite networking: A measurement-based analysis and optimization. *ACM Transactions on Multimedia Computing, Communications and Applications* 21, 9 (2025), 1–24.
- [10] Sehoon Ha, Peng Xu, Zhenyu Tan, Sergey Levine, and Jie Tan. 2021. Learning to Walk in the Real World with Minimal Human Effort. In *Proc. CoRL*. 1110–1120.
- [11] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proc. ICML*. 1861–1870.
- [12] Kevin Hsieh, Ganesh Ananthanarayanan, Peter Bodik, Shivaram Venkataraman, Paramvir Bahl, Matthai Philipose, Phillip B Gibbons, and Onur Mutlu. 2018. Focus: Querying large video datasets with low latency and low cost. In *Proc. USENIX OSDI*. 269–286.
- [13] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *Proc. ICLR*.
- [14] Jinwoo Hwang, Minsu Kim, Daeun Kim, Seungho Nam, Yoonsung Kim, Dohee Kim, Hardik Sharma, and Jongse Park. 2022. CoVA: Exploiting Compressed-Domain analysis to accelerate video analytics. In *Proc. USENIX ATC*. 707–722.
- [15] Junchen Jiang, Ganesh Ananthanarayanan, Peter Bodik, Siddhartha Sen, and Ion Stoica. 2018. Chameleon: scalable adaptation of video analytics. In *Proc. ACM SIGCOMM*. 253–266.
- [16] Xiaofan Jiang, Joseph Polastre, and David Culler. 2005. Perpetual environmentally powered sensor networks. In *Proc. ACM/IEEE IPSN*. 463–468.
- [17] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li Shiu-Peh, and Daniel Rubenstein. 2002. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with ZebraNet. In *Proc. ACM ASPLOS*. 96–107.
- [18] Aman Kansal, Jason Hsu, Sadaf Zahedi, and Mani B Srivastava. 2007. Power management in energy harvesting sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)* 6, 4 (2007), 32–es.
- [19] Simon Kassing, Debopam Bhattacharjee, André Baptista Águas, Jens Eirik Saethre, and Ankith Singla. 2020. Exploring the “Internet from Space” with Hypatia. In *Proc. ACM IMC*. 214–229.
- [20] Yoon Kim and Alexander M. Rush. 2016. Sequence-Level Knowledge Distillation. In *Proc. EMNLP*. 1317–1327.
- [21] Jiaying Li, Hao Fang, Chi Xu, Miao Zhang, Jiangchuan Liu, William I. Atlas, Katrina M. Connors, and Mark A. Spoljaric. 2026. Sustainable Intelligence for the Wild: Democratizing Ecological Monitoring via Knowledge-Adaptive Edge Expert Agents. In *Proc. IJCAI*. 7292–7300.
- [22] Yuanqi Li, Arthi Padmanabhan, Pengzhan Zhao, Yufei Wang, Guoqing Harry Xu, and Ravi Netravali. 2020. Reducto: On-camera filtering for resource-efficient real-time video analytics. In *Proc. ACM SIGCOMM*. 359–376.
- [23] Ji Lin, Jiaming Tang, Haotian Tang, Shang Yang, Wei-Ming Chen, Wei-Chen Wang, Guangxuan Xiao, Xingyu Dang, Chuang Gan, and Song Han. 2024. AWQ: Activation-aware weight quantization for on-device LLM compression and acceleration. *Proc. MLSys* 6 (2024), 87–100.
- [24] Lixin Liu, Yuanjie Li, Hewu Li, Jiabo Yang, Wei Liu, Jingyi Lan, Yufeng Wang, Jiarui Li, Jianping Wu, Qian Wu, Jun Liu, and Zeqi Lai. 2024. Democratizing Direct-to-Cell Low Earth Orbit Satellite Networks. In *Proc. USENIX NSDI*. 791–808.
- [25] Hongzi Mao, Ravi Netravali, and Mohammad Alizadeh. 2017. Neural adaptive video streaming with Pensieve. In *Proc. ACM SIGCOMM*. 197–210.
- [26] Marine Scotland Science. 2016. *Technical, Logistical, and Economic Considerations for the Development and Implementation of a Scottish Salmon Counter Network*. Scottish Marine and Freshwater Science Vol. 7, No. 2. Marine Scotland Science. <https://www.gov.scot/publications/technical-logistical-economic-considerations-development-implementation-scottish-salmon-counter-network/>
- [27] Nitinder Mohan, Andrew E Ferguson, Hendrik Cech, Rohan Bose, Prakita Rayyan Renatin, Mahesh K Marina, and Jörg Ott. 2024. A multifaceted look at Starlink performance. In *Proc. ACM WWW*. 2723–2734.
- [28] NOAA Northeast Fisheries Science Center and Greater Atlantic Regional Fisheries Office. 2020. *Electronic Monitoring Audit Model Program Reviewer Guidance Manual: Video Review Protocols for Multispecies Sector Trips*. Retrieved October 9, 2026 from https://apps-nefsc.fisheries.noaa.gov/FSBEM/index.php/docs/apidocs/Video_Reviewer_Guidance_V14.pdf
- [29] NVIDIA. 2026. *NVIDIA Jetson Orin*. Retrieved February 1, 2026 from <https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-orin/>
- [30] OpenAI. 2025. *GPT-4o*. Retrieved March 27, 2025 from <https://developers.openai.com/api/docs/models/gpt-4o?snapshot=gpt-4o-2024-11-20>
- [31] Unkyu Park and John Heidemann. 2011. Data muling with mobile phones for sensor networks. In *Proc. ACM SenSys*. 162–175.
- [32] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *Proc. ICML*. 8748–8763.
- [33] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You Only Look Once: Unified, Real-Time Object Detection. In *Proc. IEEE CVPR*. 779–788.
- [34] Yidong Ren, Amalinda Gamage, Li Liu, Mo Li, Shigang Chen, Yoonsuk Dong, and Zhichao Cao. 2024. SateRIoT: High-performance ground-space networking for rural IoT. In *Proc. ACM MobiCom*. 755–769.
- [35] Aaron Schulman, Vishnu Navda, Ramachandran Ramjee, Neil Spring, Pralhad Deshpande, Calvin Grunewald, Kamal Jain, and Venkata N Padmanabhan. 2010. Bartendr: a practical approach to energy-aware cellular data scheduling. In *Proc. ACM MobiCom*. 85–96.
- [36] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. 2024. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300* (2024).
- [37] Jayanth Shenoy, Om Chabra, Tusher Chakraborty, Suraj Jog, Deepak Vasisht, and Ranveer Chandra. 2024. CosMAC: Constellation-Aware

- Medium Access and Scheduling for IoT Satellites. In *Proc. ACM MobiCom*. 724–739.
- [38] Pieter Simoons, Yu Xiao, Padmanabhan Pillai, Zhuo Chen, Kiryong Ha, and Mahadev Satyanarayanan. 2013. Scalable crowd-sourcing of video from mobile devices. In *Proc. ACM MobiSys*. 139–152.
- [39] Jacob Sorber, Alexander Kostadinov, Matthew Garber, Matthew Brennan, Mark D Corner, and Emery D Berger. 2007. Eon: a language and runtime system for perpetual systems. In *Proc. ACM SenSys*. 161–174.
- [40] Lin Sun, Weijun Wang, Tingting Yuan, Liang Mi, Haipeng Dai, Yunxin Liu, and Xiaoming Fu. 2024. BiSwift: Bandwidth orchestrator for multi-stream video analytics on edge. In *Proc. IEEE INFOCOM*. 1181–1190.
- [41] Chen Tessler, Daniel J Mankowitz, and Shie Mannor. 2019. Reward constrained policy optimization. In *Proc. ICLR*.
- [42] Chunlin Tian, Xinpeng Qin, Kahou Tam, Li Li, Zijian Wang, Yuanzhe Zhao, Minglei Zhang, and Chengzhong Xu. 2025. CLONE: Customizing LLMs for efficient Latency-Aware inference at the edge. In *Proc. USENIX ATC*. 563–585.
- [43] Ultralytics. 2026. YOLO26. Retrieved February 1, 2026 from <https://platform.ultralytics.com/ultralytics/yolo26>
- [44] Deepak Vasisht, Jayanth Shenoy, and Ranveer Chandra. 2021. L2D2: Low latency distributed downlink for LEO satellites. In *Proc. ACM SIGCOMM*. 151–164.
- [45] Pavan Kumar Anasosalu Vasu, Fartash Faghri, Chun-Liang Li, Cem Koc, Nate True, Albert Antony, Gokul Santhanam, James Gabriel, Peter Grasch, Oncel Tuzel, et al. 2025. FastVLM: Efficient vision encoding for vision language models. In *Proc. IEEE/CVF CVPR*. 19769–19780.
- [46] Zhizhong Wan, Bin Yin, Junjie Xie, Fei Jiang, Xiang Li, and Wei Lin. 2024. LARR: Large Language Model Aided Real-time Scene Recommendation with Semantic Understanding. In *Proc. ACM RecSys*. 23–32.
- [47] Liying Wang, Qing Li, Yuhang Zhou, Zhaofeng Luo, Donghao Zhang, Shangguang Wang, Xuanzhe Liu, and Chenren Xu. 2025. Emulating space computing networks with RHONE. In *Proc. USENIX ATC*. 815–831.
- [48] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Proc. NeurIPS* 35 (2022), 24824–24837.
- [49] Dustin J Welbourne, Andrew W Claridge, David J Paull, and Andrew Lambert. 2016. How do passive infrared triggered camera traps operate and why does it matter? Breaking down common misconceptions. *Remote Sensing in Ecology and Conservation* 2, 2 (2016), 77–83.
- [50] Duo Wu, Xianda Wang, Yaqi Qiao, Zhi Wang, Junchen Jiang, Shuguang Cui, and Fangxin Wang. 2024. NetLLM: Adapting large language models for networking. In *Proc. ACM SIGCOMM*. 661–678.
- [51] Junfeng Wu, Qing-Shan Jia, Karl Henrik Johansson, and Ling Shi. 2013. Event-based sensor data scheduling: Trade-off between communication rate and estimation quality. *IEEE Trans. Automat. Control* 58, 4 (2013), 1041–1046.
- [52] Chi Xu, Rongsheng Qian, Hao Fang, Xiaoqiang Ma, William I Atlas, Jiangchuan Liu, and Mark A Spoljaric. 2024. SALINA: Towards sustainable live sonar analytics in wild ecosystems. In *Proc. ACM/IEEE SenSys*. 68–81.
- [53] Mengwei Xu, Tiantu Xu, Yunxin Liu, and Felix Xiaozhu Lin. 2021. Video analytics with zero-streaming cameras. In *Proc. USENIX ATC*. 459–472.
- [54] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. 2025. Qwen3 technical report. *arXiv preprint arXiv:2505.09388* (2025).
- [55] Qisong Yang, Thiago D Simão, Simon H Tindemans, and Matthijs TJ Spaan. 2021. WCSAC: Worst-case soft actor critic for safety-constrained reinforcement learning. In *Proc. AAAI*, Vol. 35. 10639–10646.
- [56] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2023. ReAct: Synergizing reasoning and acting in language models. In *Proc. ICLR*.
- [57] Zhangyue Yin, Qiushi Sun, Qipeng Guo, Zhiyuan Zeng, Xiaonan Li, Junqi Dai, Qinyuan Cheng, Xuan-Jing Huang, and Xipeng Qiu. 2024. Reasoning in flux: Enhancing large language models reasoning through uncertainty-aware adaptive guidance. In *Proc. ACL*. 2401–2416.
- [58] Mu Yuan, Lan Zhang, Fengxiang He, Xueting Tong, and Xiang-Yang Li. 2022. InFi: End-to-end learnable input filter for resource-efficient mobile-centric inference. In *Proc. ACM MobiCom*. 228–241.
- [59] Heng Zeng, Carla Schlatter Ellis, Alvin R Lebeck, and Amin Vahdat. 2003. Currency: A Unifying Abstraction for Expressing Energy Management Policies. In *Proc. USENIX ATC*. 43–56.
- [60] Ben Zhang, Xin Jin, Sylvia Ratnasamy, John Wawrzynek, and Edward A Lee. 2018. AWStream: Adaptive wide-area streaming analytics. In *Proc. ACM SIGCOMM*. 236–252.
- [61] Haoyu Zhang, Ganesh Ananthanarayanan, Peter Bodik, Matthai Philipose, Paramvir Bahl, and Michael J Freedman. 2017. Live Video Analytics at Scale with Approximation and Delay-Tolerance. In *Proc. USENIX NSDI*. 377–392.
- [62] Hang Zhang, Xin Li, and Lidong Bing. 2023. Video-LLaMA: An Instruction-tuned Audio-Visual Language Model for Video Understanding. In *Proc. EMNLP: System Demonstrations*. 543–553.
- [63] Miao Zhang, Jiaxing Li, Haoyuan Zhao, Linfeng Shen, and Jiangchuan Liu. 2024. StarStream: Live Video Analytics over Space Networking. In *Proc. ACM MM*. 7909–7917.
- [64] Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. 2025. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176* (2025).
- [65] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. 2022. ByteTrack: Multi-object tracking by associating every detection box. In *Proc. ECCV*. 1–21.