https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

Choosing an operational inference pipeline for internal solitary wave detection in Sentinel-1 SAR imagery: EVA02-Large+XGBoost versus SAR_CNN v2 (Lux.jl)

João Pinelo[1], Arun Shukla[1], Gilberto Titericz[2], Adriana Santos-Ferreira[1], João Gonçalves[1], João Moniz[1]

[1] *Atlantic International Research Centre (AIR Centre), Azores, Portugal*

[2] NVIDIA Inc., Curitiba, Brazil.

Corresponding author: João Pinelo, joao.pinelo@aircentre.org

Abstract. This paper presents a systematic comparative evaluation of two machine learning inference pipelines developed for the Internal Waves Service (IWS), an operational platform for the continuous automated detection of oceanic internal solitary waves (ISWs) in Sentinel-1 synthetic aperture radar (SAR) Wave mode imagery. The IWS ingests imagery from the live Sentinel-1 feed — scaling to approximately 4,000 images per day as the constellation reaches full operational capacity — and is systematically acquiring a historical archive estimated at up to 17 million images back to 2014. The two pipelines compared are a Python pipeline pairing EVA02, a 305-million-parameter pretrained vision transformer, with an XGBoost classifier; and a Julia pipeline built around a 283,329-parameter convolutional neural network implemented in Lux.jl and trained from scratch on domain-specific SAR imagery. Both pipelines were benchmarked across four deployment configurations (each on GPU and CPU) on the service's production server hardware, measuring classification accuracy, inference throughput, GPU energy consumption, and memory footprint. The Python pipeline achieves higher classification accuracy (F1 96.26 % versus 95.00 %; AUC-ROC 99.29 % versus 98.90 %), attributable to the representational capacity of the pretrained vision transformer. The Julia pipeline is 132 times faster on GPU (3,396 versus 25.6 images per second) and consumes 267 times less energy per image (43.7 versus 11,690 mJ), completing a full archive reprocessing pass in 1.4 hours versus 7.7 days. Classification is bit-for-bit identical across GPU and CPU for the Julia pipeline, confirming that the deployment target can be chosen on operational grounds without accuracy trade-offs. Per-image metrics are projected to operational volumes, quantifying annual GPU occupation (2.9 versus 384 hours at the current reprocessing cadence) and throughput headroom for future constellation expansion. Based on these findings, the IWS deploys the Julia pipeline on GPU for all inference, accepting the 1.26-percentage-point accuracy trade-off in exchange for same-day archive reclassification and minimal contention on shared institutional GPU infrastructure. The evaluation methodology — benchmarking on production hardware and projecting to operational volumes — is directly transferable to other Earth observation services evaluating inference pipeline options.

1. Introduction

Oceanic internal solitary waves (ISWs) are nonlinear gravity waves that propagate along density interfaces within the ocean interior, typically along the seasonal or permanent pycnocline (Helfrich and Melville, 2006). They can exceed 100 m in amplitude, produce vertical velocities above 0.5 m s^{-1}, and generate localised shear currents that pose measurable hazards to offshore infrastructure, submarine navigation, and underwater acoustic communication (Alford et al., 2015; Osborne and Burch, 1980). ISWs are also significant contributors to ocean mixing, energy dissipation, and vertical nutrient transport, with implications for thermohaline circulation and regional ecosystem dynamics (Garrett and Munk, 1979; Whalen et al., 2020). Despite their oceanographic and operational importance, systematic monitoring of ISWs has historically been constrained by the labour intensity of manual identification in satellite imagery and the geographic fragmentation of observational campaigns, which have concentrated on a small number of well-studied hotspots such as the South China Sea, the Amazon shelf, the Mascarene Plateau and the Strait of Gibraltar, among others (Chang et al., 2021; Jackson et al., 2012; Magalhaes et al., 2016; Roustan et al.,

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

2024; da Silva et al., 2015). Recent work has applied deep learning to automate ISW detection in SAR imagery, including object detection frameworks (Bao et al., 2020) and segmentation architectures (Zhang et al., 2023). However, these studies address algorithm development on limited datasets; none reports an operational deployment comparison at the scale of a continuous global monitoring service.

The Internal Waves Service (IWS) was developed to address this gap. Operated by the Atlantic International Research Centre (AIR Centre) and described in detail by Santos-Ferreira (Santos-Ferreira et al., 2025) and by Pinelo et al. (Pinelo et al., 2025), the IWS is the first operational, global-scale platform for the continuous automated detection of ISWs. Based on synthetic aperture radar (SAR) imagery, the service ingests Sentinel-1 Wave Mode (WV) vignettes — 20 × 20 km images acquired at 5 m resolution every 100 km along the satellite orbit — classifies each image for the presence or absence of ISW signatures using a machine learning pipeline, and routes positive detections to domain experts for validation. The validated detections accumulate into a growing, curated dataset that feeds subsequent model retraining, creating a continuous improvement cycle. At its design operational scale, the IWS is dimensioned to process approximately 4,000 new vignettes per day from the live Sentinel-1 feed once the full constellation is operational, with historical reprocessing campaigns systematically ingesting the archive back to the mission start in 2014 — an estimated corpus of up to 17 million WV mode acquisitions.

The computational demands of this service are substantial, continuous, and growing. The Sentinel-1 constellation is currently evolving, with Sentinel-1C (launched in December 2024) now operational and contributing to data acquisition, while Sentinel-1D (launched in November 2025) is currently undergoing commissioning and is expected to become operational in the near future. Sentinel-1A continues to operate beyond its nominal design lifetime and remains an important contributor to the current observation capacity, following the loss of Sentinel-1B in 2022. Once fully operational, the expanded Sentinel-1 constellation is expected to enhance revisit frequency and increase WV mode acquisition volume relative to single-satellite operations (Torres et al., 2012). Future missions such as ROSE-L (expected 2028) will further increase the flow of SAR imagery likely suitable for ISW detection. An operational inference pipeline that will run for years at this scale must satisfy multiple, potentially competing requirements simultaneously: classification accuracy sufficient to support scientific analysis, throughput sufficient to process both the daily feed and the growing historical archive within practical time horizons, and deployment feasibility on the available institutional infrastructure.

Two candidate inference pipelines have been developed for the IWS, embodying fundamentally different design philosophies. Both are trained on IWS expert-validated SAR imagery, but they differ in how they approach the classification problem. The first is a Python-based pipeline that pairs EVA02 — a 305-million-parameter vision transformer pretrained on approximately 38 million images via masked image modelling (Fang et al., 2023) — with an XGBoost gradient-boosted decision tree classifier (Chen and Guestrin, 2016) trained on the IWS dataset. This pipeline leverages large-scale transfer learning: EVA02 extracts rich, general-purpose visual embeddings that XGBoost then maps to the ISW classification decision. The second is a Julia-based pipeline built around a purpose-designed convolutional neural network (CNN) of approximately 283,000 parameters, implemented in the Lux.jl differentiable programming framework (Pal, 2023, 2026) and trained end-to-end from scratch on IWS-labelled SAR imagery. This pipeline trades the representational capacity of a pretrained foundation model for a lightweight, self-contained architecture with no external dependencies beyond the Julia runtime.

These two pipelines occupy different positions in the design space not because one is a refinement of the other, but because they reflect distinct engineering strategies for the same operational problem. The Python pipeline prioritises classification accuracy through transfer learning at the cost of computational weight. The Julia pipeline prioritises inference efficiency through architectural simplicity at the cost of starting without pretrained knowledge. The question facing the IWS is not which pipeline is abstractly "better," but which pipeline, deployed on which hardware, best serves the long-term operational requirements of the service — and at what cost.

This paper presents a systematic comparative evaluation of these two pipelines across four deployment configurations: each pipeline running inference on GPU and on CPU, all on identical server hardware. We measure classification accuracy (AUC-ROC, F1 score, precision, recall, and threshold sensitivity), inference throughput (images per second), GPU energy consumption (millijoules per image), and memory footprint (peak GPU VRAM and CPU RAM). From the throughput measurements, we derive operational projections at the processing volumes of the IWS, quantifying time-to-solution, GPU opportunity cost on shared infrastructure, and throughput headroom for future constellation expansion.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

The contribution of this paper is threefold. First, it provides the evidence base for an operational deployment decision — one that we make explicit in the Discussion — rather than an open-ended comparison. Second, it demonstrates that the choice of inference pipeline has consequences for throughput and GPU occupation that are large enough to warrant the same systematic analysis typically reserved for model accuracy. For operational Earth observation services processing imagery continuously at global scale, the inference pipeline is infrastructure: it runs for years, and its compute costs accumulate. Third, it contributes to the emerging literature on responsible deployment of machine learning in operational scientific settings, where long-term sustainability and institutional maintainability matter alongside predictive performance.

The remainder of this paper is organised as follows. Section 2 describes the IWS platform architecture, data ingestion, expert validation loop, and dataset construction. Sections 3 and 4 describe the two candidate pipelines: the Python EVA02+XGBoost pipeline and the Julia Lux CNN pipeline, respectively. Section 5 presents the comparative evaluation across four deployment configurations (each pipeline on GPU and CPU), covering classification accuracy, inference throughput, energy consumption, and memory footprint. Section 6 projects the per-image benchmark results to the operational volumes of the IWS, quantifying time-to-solution, GPU opportunity cost, and throughput headroom for future constellation expansion. Section 7 discusses the deployment decision, limitations, and generalisability. Section 8 concludes.

2. The Internal Waves Service

2.1 System architecture and data ingestion

The Internal Waves Service (IWS) is an operational platform for the continuous automated detection of internal solitary waves (ISWs) in Sentinel-1 Wave (WV) mode synthetic aperture radar (SAR) imagery. The service is designed around a continuous processing loop: raw imagery is ingested from the ESA Copernicus Dataspace, classified by a machine learning pipeline, referred to domain experts for validation, and returned as curated, labelled data that feeds subsequent model retraining (Figure 1).

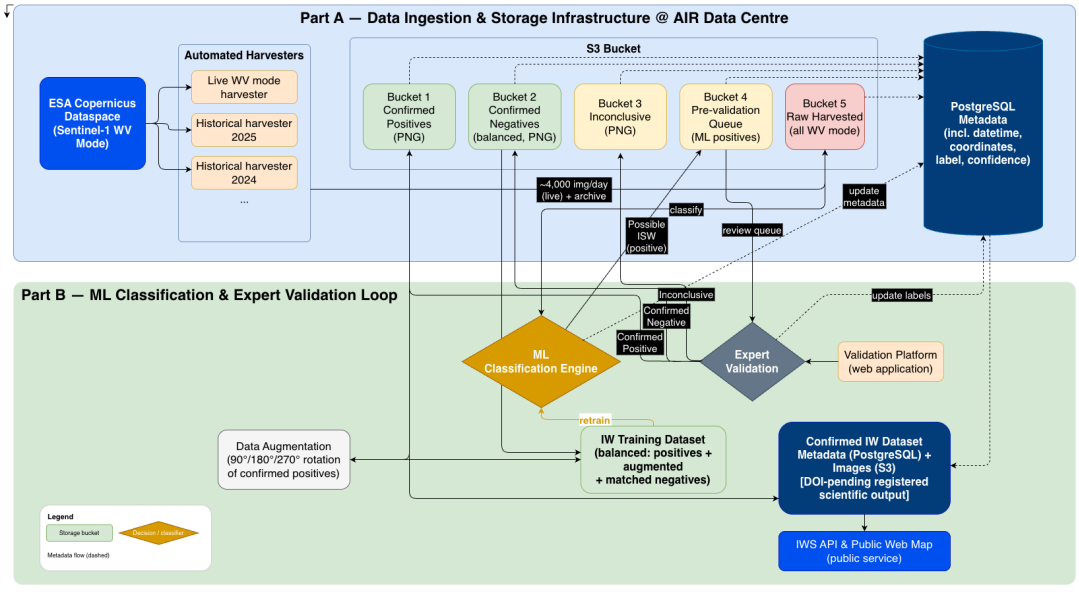

Figure 1. Architecture of the Internal Waves Service (IWS). Part A: automated harvesters ingest Sentinel-1 WV mode vignettes from the ESA Copernicus Dataspace into S3 storage, with metadata managed in PostgreSQL. Part B: the ML classification engine routes predicted positives to domain experts for validation. Confirmed labels feed a balanced training dataset through rotational augmentation of positives and matched negatives, closing a continuous

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

retraining loop. The validated dataset is issued with a persistent DOI and served via the IWS API and public web map.

Data ingestion is performed by automated harvesters operating in parallel. A live harvester runs continuously, downloading newly acquired WV mode vignettes as they become available through the Copernicus Dataspace API. A complementary set of historical harvesters processes the Sentinel-1 archive systematically by year, enabling retrospective coverage back to the mission start in 2014. All ingested images are stored as PNG files in an S3-compatible object store (AIR Data Centre), organised by acquisition date and geographic location. Associated metadata — including acquisition datetime, image bounding coordinates, and classification status — are managed in a PostgreSQL relational database, providing efficient query access across the full dataset.

The inference pipeline that processes this imagery is long-lived infrastructure: the choice of model architecture and deployment hardware determines the service's throughput capacity and operational cost for years to come. As the archive grows — estimated at up to 17 million images once historical ingestion is complete, with the daily feed scaling to approximately 4,000 images as the full Sentinel-1 constellation becomes operational — the cost of each model iteration's archive reprocessing grows in step.

2.2 Classification and expert validation loop

Each ingested image is passed to the classification pipeline, which assigns a binary prediction — internal wave present or absent — along with a confidence score. Images classified as potential positive detections are forwarded to a purpose-built web-based validation platform, where domain experts in physical oceanography review each vignette and assign one of three labels: Confirmed Positive (ISW signal present), Confirmed Negative (no ISW detected), or Inconclusive (classification not possible from the vignette alone). The Inconclusive category captures cases where SAR signatures are ambiguous, partially resolved, or potentially confounded by turbulence or other oceanic surface phenomena.

The decision to route only ML-predicted positives to expert review is operationally motivated. Manual review of all ingested images — thousands per day — is not feasible. The classification pipeline therefore functions as a pre-screening filter, and the expert validation step serves both to confirm detections and to generate high-quality labelled data for subsequent retraining. This human-in-the-loop architecture means that classification errors propagate asymmetrically: false positives entering the validation queue incur reviewer time but are corrected before entering the dataset, whereas false negatives — ISW events rejected by the classifier without expert review — represent permanent data loss. This asymmetry motivates the recall-biased operating threshold discussed in Section 5.

False positives identified during expert review are retained as confirmed negative samples rather than discarded. These are particularly valuable training examples because they represent the decision boundary cases most likely to challenge the classifier — oceanic patterns visually similar to ISW signatures that the model incorrectly elevated to the validation queue.

2.3 Dataset construction and class balance

The labelled dataset accumulates continuously as validation proceeds. Confirmed positive samples are augmented through 90°, 180°, and 270° rotations, quadrupling the effective positive count without introducing artificial features; ISW surface signatures in WV mode imagery have no preferred orientation relative to the satellite look direction, making rotational augmentation physically appropriate. Confirmed negatives are sampled to maintain approximate class balance with the augmented positive set; excess ML-rejected negatives are not retained.

At the time of writing, the labelled dataset comprises 2500 confirmed positive detections, approximately 1700 confirmed negatives, and 4640 images labelled as Inconclusive (dataset statistics as of March 25, 2026). The global spatial distribution of confirmed detections is shown in Figure 2.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

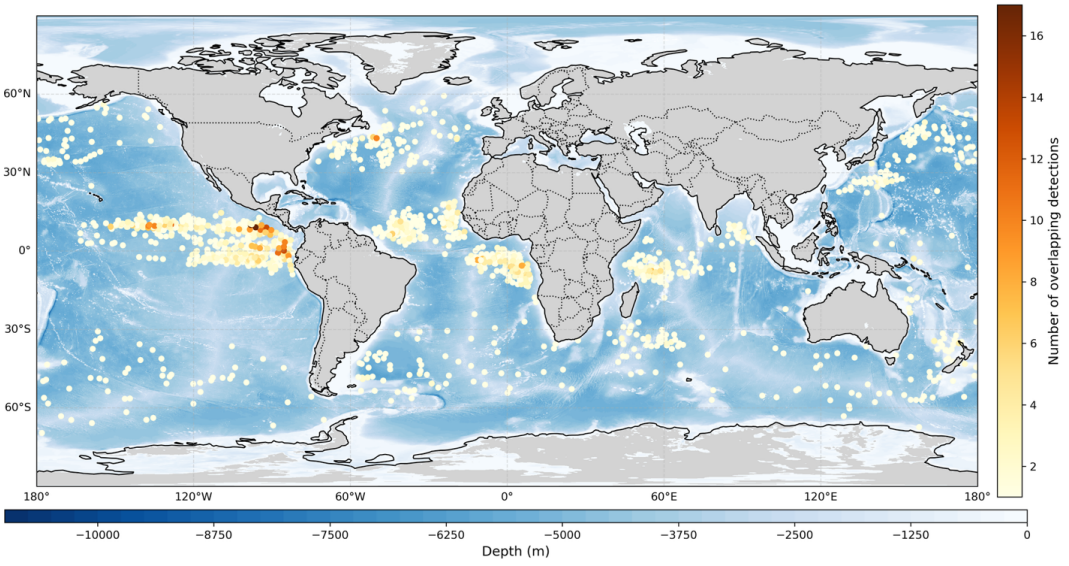

Figure 2. Global distribution of validated ISW detections from Sentinel-1 WV mode imagery. Dots mark centroids of ISW events detected since June 2024, with a 5-years extended dataset for the equatorial Pacific (September 2020-25). The color scale represents the number of overlapping detections at each location, with darker shades indicating higher recurrence of ISW activity in the same region.

The model is retrained periodically as the dataset grows. Each retraining cycle uses a stratified train-test split, with class balance maintained across both partitions. Upon retraining, the entire archive will be reclassified with the new model and a new version of the dataset is issued with a persistent DOI, ensuring that each published version is internally consistent — every image classified by the same model. The test partition used for all comparisons in this paper — 5,860 images comprising exactly 2,930 confirmed positives and 2,930 confirmed negatives (including augmentation) — is held out from all training runs. This perfect 1:1 class balance eliminates the need for class-weighted metrics and makes accuracy directly interpretable alongside F1 and AUC-ROC.

Two candidate inference pipelines have been developed for the IWS, representing fundamentally different approaches to the same classification task. Both operate within the service architecture described above; they differ in model architecture, computational requirements, and deployment characteristics. Sections 3 and 4 describe each pipeline in turn. Section 5 evaluates both pipelines across four deployment configurations — each on GPU and CPU — to provide the evidence base for the operational deployment decision that is the purpose of this paper.

3. Pipeline A: Python EVA02+XGBoost

3.1 Context and development

The Python pipeline was the first classification system deployed operationally in the IWS. Its development followed a two-stage approach. An initial proof-of-concept was developed in-house at the AIR Centre, demonstrating that machine learning classification of ISW signatures in WV mode imagery was feasible and validating the dataset, labelling protocol, and evaluation methodology. With the approach validated, a Kaggle machine learning competition was organised, providing the community with a labelled subset of the IWS dataset and inviting participants to develop a binary classifier for ISW detection. The winning solution, developed by Gilberto Titericz, formed the basis of the operational pipeline described here.

The pipeline's contribution to the IWS extends beyond its classification performance. Through its continued operational deployment and the expert validation loop described in Section 2, it generated the growing labelled

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

dataset that underpins all subsequent model development — including the Julia pipeline evaluated alongside it in this paper.

3.2 Architecture

The pipeline implements a two-stage inference approach combining a large pretrained feature extractor with a lightweight classifier.

In the first stage, each WV mode vignette is converted to an RGB image, resized to 448 × 448 pixels, standardised, and passed through EVA02 — a vision transformer from the Timm library (Wightman et al., 2023) using the eva02_large_patch14_448.mim_m38m_ft_in1k checkpoint. EVA02-Large is a 305-million-parameter model pretrained on the Merged-38M dataset — a combination of ImageNet-22K, CC12M, CC3M, COCO, ADE20K, Object365, and OpenImages, totalling approximately 38 million images — using masked image modelling with EVA-CLIP as the teacher model (Fang et al., 2023). The model was subsequently fine-tuned on ImageNet-1K. The architecture employs mean pooling, SwiGLU activations, and Rotary Position Embeddings (RoPE). A 1,000-dimensional feature embedding is extracted from the model's softmax classification head, capturing rich visual representations transferable to the ISW detection task.

In the second stage, the extracted embeddings are passed to an XGBoost gradient-boosted decision tree classifier (Chen and Guestrin, 2016), trained on the IWS dataset, which produces the final binary prediction and confidence score. XGBoost is configured with a histogram-based tree method, a learning rate of 0.05, restricted tree depth, and row and feature subsampling of 50% and 75% respectively per tree.

The hybrid architecture was motivated empirically. An initial MLP classifier developed in-house and applied directly to raw image features achieved a baseline accuracy of 58.2% (AUC-ROC = 0.60), confirming that shallow feature representations were insufficient for the task. A naive unoptimised CNN applied to the same data achieved 50.7% accuracy (AUC-ROC = 0.50) — equivalent to random classification on a balanced dataset. The introduction of EVA02 embeddings as the feature representation raised classifier performance to the levels reported in Section 5, a gain attributable to EVA02's pretrained representational capacity capturing texture and structural features relevant to ISW surface signatures that simpler representations could not resolve.

3.3 Inference pipeline

At inference time, the full pipeline executes both stages sequentially for every input image: EVA02 embedding extraction followed by XGBoost classification. On GPU, both stages execute end-to-end; on CPU, the same pipeline runs without CUDA acceleration. In both cases, the EVA02 embedding step dominates inference cost — it requires a full forward pass through a 305-million-parameter transformer for every input image. The XGBoost classification step is negligible by comparison: embedding extraction accounts for approximately 99.7% of per-image compute in the combined pipeline.

The pipeline's software dependencies include PyTorch, the Timm model library, and XGBoost, with EVA02's pretrained weights downloaded from the Hugging Face model hub. The pretrained weights are frozen at inference time; only the XGBoost classifier was trained on IWS data.

3.4 Operational role

The Python pipeline was deployed operationally and has processed incoming WV mode imagery since March 2025, routing ML-predicted positives to the expert validation platform. Direct ingestion from the ESA Copernicus Dataspace replaced the previous intermediary data source, providing more reliable access to the full global WV mode acquisition stream.

The pipeline's throughput is dominated by the EVA02 embedding stage. At the design operational scale of approximately 4,000 new acquisitions per day, the pipeline's throughput on available hardware is sufficient for same-day processing of the live feed. However, the computational cost of EVA02 inference imposes a significant constraint on archive reprocessing — where the full historical archive (estimated at up to 17 million images) requires classification with each major model iteration — and on readiness for the increased acquisition volumes expected from future constellation expansion. The quantitative throughput and energy characteristics of this pipeline, measured on the production server hardware, are reported in Section 5.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

4. Pipeline B: Julia Lux CNN

4.1 Motivation and design constraints

The Julia pipeline was developed as a purpose-built alternative to the Python EVA02+XGBoost pipeline described in Section 3, motivated by three operational considerations. First, the EVA02 embedding stage dominates the Python pipeline's inference cost: a 305-million-parameter vision transformer must execute a full forward pass for every input image, even though the downstream classification decision is made by a comparatively lightweight XGBoost model. For a service processing thousands of images daily and reprocessing an archive estimated at up to 17 million images with each model iteration, the per-image cost of the embedding stage accumulates into a substantial compute burden. Second, the Python pipeline's dependency on a large pretrained model creates a coupling between the IWS and the external model ecosystem — EVA02's weights, input preprocessing conventions, and software dependencies must be maintained across the service's operational lifetime. Third, the IWS training dataset is domain-specific and growing: every validated detection or rejection adds to the corpus of expert-labelled SAR imagery available for model training. A lightweight architecture trained from scratch on this dataset can improve in step with the data, without requiring re-extraction of embeddings through a frozen foundation model.

The design objective was therefore a self-contained inference pipeline with the smallest feasible model that achieves operationally adequate classification performance on WV mode SAR vignettes, implemented in a high-performance language suitable for scientific computing. Julia (Bezanson et al., 2017) was selected for its combination of high-level expressiveness and compiled performance, its native GPU support through CUDA.jl, and its growing ecosystem for differentiable programming and Earth observation (Lux.jl, Flux.jl, JuliaEO).

4.2 Model architecture

The classifier is a convolutional neural network (SAR_CNN) implemented in the Lux.jl differentiable programming framework(Pal, 2026). The architecture comprises four convolutional feature extraction blocks followed by global average pooling and a two-layer dense classifier head. Input images are single-channel (greyscale) SAR vignettes resized to 256 × 256 pixels and normalised to zero mean and unit variance using dataset-derived statistics (μ = 0.380, σ = 0.156).

Table 1 summarises the architecture. Each convolutional block consists of a convolution, batch normalisation, ReLU activation, optional 2 × 2 max-pooling, and spatial dropout. The first block applies a 7 × 7 convolution with stride 2, halving the spatial dimensions and expanding from 1 to 32 channels. Subsequent blocks use 5 × 5 and 3 × 3 kernels with same-padding, progressively widening the channel dimension to 128 while max-pooling reduces spatial resolution by a factor of 2 at each stage. The final convolutional output is a 16 × 16 × 128 feature map, which global average pooling compresses to a 128-dimensional vector. The classifier head maps this vector through a 128 → 64 dense layer with ReLU activation, dropout, and a 64 → 1 output layer with sigmoid activation producing a scalar confidence score in (0, 1).

Table 1. SAR_CNN v2 architecture. Each ConvBlock applies Conv → BatchNorm → ReLU → MaxPool (where indicated) → spatial dropout. Input: single-channel 256 × 256 px SAR vignette. Total trainable parameters: 283,329.

Block	Kernel	Channels	Stride	Pooling	Dropout	Output shape	Parameters
ConvBlock 1	7 × 7	1 → 32	2	—	0.05	128 × 128 × 32	1,664
ConvBlock 2	5 × 5	32 → 64	1	2 × 2	0.10	64 × 64 × 64	51,392
ConvBlock 3	3 × 3	64 → 128	1	2 × 2	0.15	32 × 32 × 128	74,112
ConvBlock 4	3 × 3	128 → 128	1	2 × 2	0.20	16 × 16 × 128	147,840
GlobalAvgPool	—	—	—	—	—	128	0
Dense 1	—	128 → 64	—	—	0.40	64	8,256
Dense 2	—	64 → 1	—	—	—	1	65
Total							**283,329**

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

The model is 1,076× smaller than EVA02 by parameter count (283,329 versus 305 million). This size difference is the primary driver of the throughput advantages reported in Section 5. The associated accuracy deficit is modest — 1.26 percentage points F1 — and reflects the trade-off between the Julia CNN learning visual representations from the IWS training dataset alone, whereas EVA02 arrives with representations pretrained on 38 million images spanning thousands of categories. As discussed in Section 7.3, this gap is expected to narrow as the IWS dataset grows.

4.3 Regularisation and training

The architecture employs an aggressive regularisation strategy appropriate to its small parameter count and the size of the available training dataset. Spatial dropout rates increase progressively through the convolutional blocks (0.05 → 0.10 → 0.15 → 0.20), with the classifier head applying 0.40 dropout. Label smoothing with $\alpha = 0.1$ replaces hard binary targets (0, 1) with softened values (0.05, 0.95), reducing the model's incentive to produce overconfident predictions at the cost of a modest calibration shift.

Training used the AdamW optimiser with a one-cycle learning rate schedule (base learning rate 1×10^{-4}, peak 5×10^{-4}) and weight decay of 0.01, for 100 epochs with a batch size of 128 (micro-batch size 64 for gradient accumulation). The model was trained on NVIDIA GPU hardware using CUDA, with the best checkpoint selected by validation loss. The checkpoint used for all benchmarks in this paper (model_v2, trained 2 March 2026) achieved a validation loss of 0.145 and a TTA-augmented validation accuracy of 95.7%.

4.4 Inference pipeline

At inference time, each SAR vignette is loaded, resized to 256 × 256 pixels, converted to a single-channel Float32 tensor, and normalised using the dataset statistics embedded in the model checkpoint ($\mu = 0.380$, $\sigma = 0.156$). Images are batched (batch size 128) and passed through the CNN in a single forward pass with dropout disabled. The sigmoid output is compared against a decision threshold to produce the binary classification. The entire inference pipeline — from image loading through classification — is implemented in Julia with no Python interoperability layer, no external model weights to download or manage, and no preprocessing steps beyond resizing and normalisation.

The same model checkpoint and architecture code are used for both GPU and CPU deployment. On GPU, the model weights and input tensors are transferred to device memory via CUDA.jl (LuxCUDA); on CPU, inference uses Julia's BLAS-accelerated linear algebra with multi-threaded matrix operations. No modifications to the model architecture or forward pass are required to switch between hardware targets.

5. Comparative Evaluation

5.1 Experimental design

Hardware

All benchmarks were conducted on a server at the AIR Data Centre, Terceira Island, Azores. The server is equipped with 2× Intel Xeon Gold 5420+ processors (28 cores each, 2.0 GHz base / 4.1 GHz boost, SMT disabled, 56 cores total), 502 GB DDR4 RAM, and an NVIDIA L40 GPU (46 GB GDDR6 ECC, 300 W TDP) running CUDA 13.1 under driver version 590.48.01. Benchmarks ran inside a containerised environment (Oracle Linux 9.7) on a Rocky Linux 10.1 host (kernel 6.12.0-124.28.1.el10_1.x86_64), with full access to all 56 CPU cores, 502 GB RAM, and the L40 GPU. GPU clocks were not locked; both GPU pipelines ran under identical default boost profiles (max graphics clock 2,490 MHz, max memory clock 9,001 MHz). This reflects the intended operational configuration: the IWS runs under default boost behaviour, and the benchmarks measure performance as it would be experienced in production.

Test set

The evaluation dataset comprised 5,860 images with a perfect 1:1 class balance — 2,930 confirmed positive (ISW present) and 2,930 confirmed negative (ISW absent) samples — drawn from the IWS expert-validated dataset described in Section 2.3. This balanced partition eliminates the need for class-weighted metrics and makes accuracy

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

directly interpretable alongside F1 and AUC-ROC. The test set was held out from all training runs for both pipelines.

Pipelines evaluated

Four deployment configurations were benchmarked, representing the full 2 × 2 combination of pipeline and hardware target:

1. **Pipeline A, GPU** — Python EVA02+XGBoost on NVIDIA L40. Full pipeline: each image loaded from disk, resized to 448 × 448 pixels, passed through EVA02 (305M parameters, loaded from timm with torch.compile applied for graph-level optimisation), and classified by a pre-trained XGBoost model. Two warm-up passes were executed before measured runs to absorb torch.compile JIT compilation overhead.
2. **Pipeline B, GPU** — Julia Lux CNN (SAR_CNN v2, 283,329 parameters) on NVIDIA L40. Full inference pipeline, no TTA. Checkpoint: model_v2 (trained 2 March 2026, normalisation $\mu = 0.380$, $\sigma = 0.156$). Three warm-up passes were executed before measured runs to ensure full Julia JIT compilation.
3. **Pipeline A, CPU** — Python EVA02+XGBoost on CPU. Full pipeline: EVA02 embedding + XGBoost classification end-to-end from raw images. No pre-cached embeddings.
4. **Pipeline B, CPU** — Julia Lux CNN (SAR_CNN v2) on CPU. Same model checkpoint as (2), same architecture. No TTA.

All pipelines used a batch size of 128. The Julia GPU and Julia CPU pipelines used the same model checkpoint and identical architecture code (adapted for CPU execution by removing CUDA device transfers). No Test-Time Augmentation was applied to any primary benchmark.

Measurement protocol

Each pipeline was profiled over five independent runs. Results are reported as mean ± standard deviation across these runs.

Throughput and latency. Wall-clock inference time was measured from the start of the first batch to the completion of the last batch, excluding model loading, data preloading, and warm-up. Throughput is reported as images per second; latency as milliseconds per image.

Classification accuracy. AUC-ROC was computed from continuous model outputs across the full test set. F1 score, accuracy, precision, and recall are reported at the optimal decision threshold (maximising F1) and at the default threshold of 0.5. A threshold sweep from 0.30 to 0.70 in steps of 0.01 is provided for each pipeline.

Energy consumption (GPU pipelines). GPU power was sampled via nvidia-smi synchronous polling at 100 ms intervals during the inference loop only. A 60-second idle baseline was recorded before each pipeline to establish quiescent power draw. Gross energy per image was computed as mean GPU power (W) × mean wall-clock time (s) / number of images. Net energy per image subtracts the idle baseline. Energy is reported for the GPU pipelines only; the CPU configurations' archive-reprocessing times (12.9 to 213 days) make them operationally noncompetitive for the IWS's primary workloads, limiting the practical relevance of CPU energy profiling.

Memory footprint. Peak GPU VRAM was recorded via CUDA memory allocation queries (Julia) or torch.cuda.max_memory_allocated() (Python). Peak CPU RAM was recorded via process-level resident set size monitoring.

Cooldown. A minimum 30-second cooldown separated consecutive profiling runs; at least 5 minutes separated runs of different pipelines. No other compute workloads ran in the container during benchmarking.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

5.2 Results

Classification accuracy

Table 2 presents the classification performance of each pipeline across both hardware targets. Since accuracy is determined by the model and its weights — not by the hardware executing it — each pipeline produces identical (Julia) or near-identical (Python) results on GPU and CPU.

Table 2. Classification accuracy across four deployment configurations.

Metric	Pipeline A GPU	Pipeline A CPU	Pipeline B GPU	Pipeline B CPU
AUC-ROC (%)	99.29	99.29	98.90	98.90
F1 at optimal threshold (%)	96.26	96.24	95.00	95.00
Accuracy at optimal threshold (%)	96.23	96.25	95.00	95.00
Precision at optimal threshold	0.954	0.964	0.950	0.950
Recall at optimal threshold	0.972	0.961	0.951	0.951
Optimal threshold	0.48	0.64	0.47	0.47
F1 at threshold = 0.5 (%)	96.24	96.22	94.70	94.70

The Julia pipeline produces bit-for-bit identical results on GPU and CPU: AUC-ROC 98.90%, F1 95.00%, and an identical confusion matrix (TP = 2,785, TN = 2,782, FP = 148, FN = 145) at threshold 0.47 (Figure 3). This confirms that the same checkpoint was used on both hardware targets and that no numerical divergence was introduced by the hardware transfer.

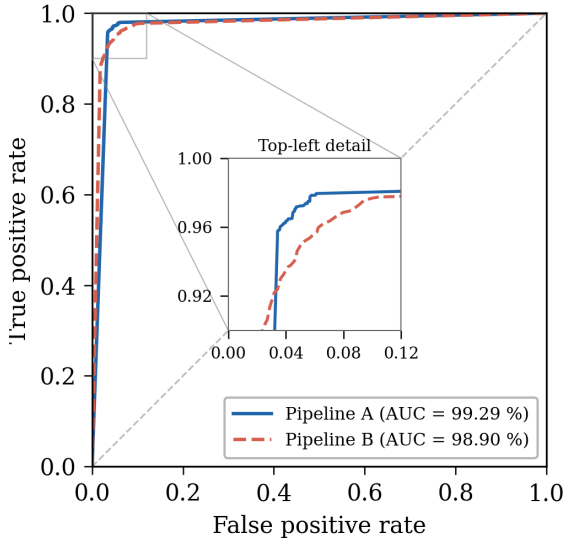

Figure 3. ROC curves for Pipeline A (Python EVA02+XGBoost, AUC = 99.29 %) and Pipeline B (Julia Lux CNN, AUC = 98.90 %) on the shared 5,860-image test set (2,930 positives, 2,930 negatives). Inset: detail of the top-left corner showing the region where the two curves diverge. Threshold sweep: 0.30–0.70 in 0.01 steps.

The Python pipeline produces near-identical results across hardware targets (AUC-ROC difference of 0.002 percentage points), with minor differences attributable to floating-point arithmetic between GPU and CPU execution paths. The different optimal thresholds reported for Python GPU (0.48) and Python CPU (0.64) reflect a flat F1 landscape rather than a meaningful accuracy difference — F1 varies by less than 0.03 percentage points across the 0.42–0.68 range (Figure 3).

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

Pipeline A achieves higher accuracy than Pipeline B across all metrics: +0.39 percentage points AUC-ROC, +1.26 percentage points F1, and a notably lower false negative count (83 vs 145 at optimal threshold on GPU). In the IWS context, where false negatives represent permanent loss of ISW detections from the scientific record, this difference is operationally relevant. However, as discussed in Section 7.2, the accuracy advantage must be weighed against the throughput and computational cost required to achieve it. Pipeline B represents the first iteration of the Julia CNN; with continued dataset growth and model development, subsequent versions are expected to close this gap.

Inference throughput

Table 3. presents the throughput and computational resource usage of each deployment configuration.

Table 3. Inference throughput and resource usage.

Metric	Pipeline A GPU	Pipeline B GPU	Pipeline A CPU	Pipeline B CPU
Throughput (img s^{-1})	25.6	3,396	0.92	15.2
Latency (ms img^{-1})	39.0	0.29	1,084	65.8
Wall time, 5,860 images (s)	228.5 ± 1.0	1.7 ± 0.06	6,354 ± 23	388 ± 34
Peak GPU VRAM (MB)	6,423	27,839	—	—
Peak CPU RAM (MB)	9,326	4,212	4,020	3,026

Pipeline B GPU is the fastest configuration by a wide margin, processing the 5,860-image test set in 1.7 seconds at 3,396 images per second — 132× faster than Pipeline A GPU (25.6 images per second) and 223× faster than its own CPU counterpart. The throughput hierarchy across all four configurations spans nearly four orders of magnitude: Pipeline B GPU (3,396) → Pipeline A GPU (25.6) → Pipeline B CPU (15.2) → Pipeline A CPU (0.92 images per second).

The throughput difference between the two GPU pipelines reflects both the architectural asymmetry (283,329 vs 305 million parameters) and the nature of the computation. Pipeline A's EVA02 forward pass dominates its inference time — at approximately 4.9 seconds per batch of 128 images on the L40, the 305-million-parameter vision transformer saturates the GPU (mean power draw 299.8 W, effectively at the 300 W TDP ceiling). Pipeline B's lightweight CNN completes the same batch in approximately 35 milliseconds, leaving substantial GPU headroom (mean power draw 153 W, 51% of TDP).

On CPU, Pipeline B is 16.5× faster than Pipeline A, a smaller ratio than on GPU (132×). This is expected: the L40 GPU's massively parallel architecture disproportionately accelerates the larger model's matrix operations, whereas on CPU the throughput ratio more closely tracks the parameter count ratio (1,076×) moderated by memory hierarchy and BLAS optimisation effects.

Peak VRAM usage is inverted relative to model size: Pipeline B uses 27.8 GB versus Pipeline A's 6.4 GB. This is attributable to Julia's CUDA memory pre-allocation strategy rather than to model weight storage (the Julia CNN's 283K parameters occupy approximately 1.1 MB at Float32). CPU RAM usage is more proportionate, with Pipeline A requiring 2–3× more than Pipeline B across both hardware targets.

GPU energy consumption

Table 4 presents energy metrics for the GPU pipelines. CPU energy was not measured as their throughputs are less operationally relevant, demoting Pipeline B's to emergency fallback (Section 5.1).

Table 4. GPU energy consumption per image.

Metric	Pipeline A GPU	Pipeline B GPU	Ratio (A/B)
Mean GPU power (W)	299.8 ± 1.8	153.0 ± 8.0	2.0×
Idle baseline (W)	78.5	77.5	—
Gross energy/image (mJ)	11,690	43.7	267×
Net energy/image (mJ)	8,630	23.6	366×

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

Metric	Pipeline A GPU	Pipeline B GPU	Ratio (A/B)
GPU temperature (°C)	64–65	38–39	—
Thermal throttling	No	No	—

Pipeline A consumes 267× more energy per image than Pipeline B (gross) and 366× more (net, subtracting idle power). The energy difference exceeds the throughput difference (132×) because Pipeline A also draws nearly twice the instantaneous power: 300 W (at the L40's TDP ceiling) versus 153 W (51% of TDP). The combination of longer inference time and higher power draw produces a compounding effect on energy consumption.

The net-to-gross ratio differs markedly between pipelines. Pipeline B's net energy (23.6 mJ) is 54% of its gross energy (43.7 mJ), meaning that idle power accounts for nearly half the GPU's draw during Julia inference — the computation is so brief that the GPU spends most of each measurement window near idle. Pipeline A's net energy (8,630 mJ) is 74% of gross (11,690 mJ), reflecting sustained full-load operation where idle overhead is proportionally small.

Neither pipeline triggered thermal throttling. Pipeline A raised the GPU to 64–65°C, well within the L40's operating envelope but notably warmer than Pipeline B's 38–39°C — barely above the 33°C idle baseline.

6. Operational Impact

The per-image metrics presented in Section 5 characterise each pipeline in isolation. This section projects those metrics to the operational volumes of the IWS and the infrastructure context of the AIR Data Centre, translating benchmark results into quantities that inform a deployment decision: time-to-solution, GPU availability, and throughput headroom for future constellation expansion.

6.1 Processing volumes and reprocessing cadence

Once the full Sentinel-1 constellation reaches operational capacity, the IWS is dimensioned to process approximately 4,000 new WV mode vignettes per day from the live feed. The full Sentinel-1 WV mode archive back to 2014 is estimated at up to 17 million images. Both the 4,000 images per day and 17 million image figures are used as planning parameters throughout Section 6 to dimension the operational projections. Each major model retrain requires a complete reprocessing pass over this archive to produce a new, internally consistent version of the dataset issued with a persistent DOI. At the current retraining cadence of approximately twice per year (R = 2), the annual inference workload comprises the continuous daily feed plus two full archive passes. If the cadence increases to quarterly reprocessing (R = 4), the archive component doubles.

6.2 Time-to-solution and GPU occupation

Table 5 projects the per-image throughput from Section 5 to the two operational tasks: the daily live feed and a single archive reprocessing pass.

Table 5. Time-to-solution at IWS operational volumes.

	Pipeline A GPU	Pipeline B GPU	Pipeline A CPU	Pipeline B CPU
Daily feed (4,000 images)	2.6 min	**1.2 s**	1.2 hours	4.4 min
Full archive (17 M images)	7.7 days	**1.4 hours**	213 days	12.9 days

For the daily feed, all configurations except Pipeline A CPU complete within minutes — throughput is unconstrained. The differences are operationally meaningful only for archive reprocessing, where they span four orders of magnitude. Pipeline B GPU completes a full archive pass in 1.4 hours; Pipeline A GPU requires 7.7 days. This is the difference between issuing a new dataset version the same afternoon and waiting over a week, during which the GPU is unavailable for other institutional workloads.

Pipeline A CPU is operationally non-competitive for archive reprocessing: at 213 days per pass, a single reprocessing cycle would not complete before the next retrain. Pipeline B CPU, at 12.9 days, is slow but feasible for a quarterly cadence and could serve as a GPU-free fallback for the daily feed (4.4 minutes).

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

Table 6 translates time-to-solution into annual GPU occupation — the resource that determines what else the shared L40 can do.

Table 6. Annual GPU occupation (hours).

	Pipeline A GPU	Pipeline B GPU
Daily feed (hours/year)	15.8	0.12
Archive × 2 (hours/year)	368	2.8
Archive × 4 (hours/year)	737	5.6
Total at R = 2	**384**	**2.9**
Total at R = 4	**753**	**5.7**

At R = 2, Pipeline A GPU occupies the L40 for 384 hours per year — 4.4% of the available 8,760 hours. Pipeline B GPU occupies it for 2.9 hours — 0.03%. At R = 4, Pipeline A rises to 753 hours (8.6%), while Pipeline B remains under 6 hours. On shared infrastructure where the GPU also serves MOHID ocean modelling, atmospheric simulations, and development workloads, this difference in opportunity cost is the primary operational consideration.

Following the "Green AI" principle that efficiency should be reported alongside accuracy (Schwartz et al., 2020), we report gross energy consumption per image for both GPU configurations. The throughput advantage of Pipeline B GPU is achieved at lower, not higher, energy cost per image. Pipeline B consumes 43.7 mJ per image at a mean GPU power draw of 153 W; Pipeline A consumes 11,690 mJ per image at 300 W, near the L40's thermal design power. The 267× energy ratio exceeds the 132× throughput ratio because Pipeline A draws nearly twice the instantaneous power while taking 132× longer per image — the two factors compound. At operational volumes, the annual GPU energy difference is modest in absolute terms (115 kWh for Pipeline A versus 0.4 kWh for Pipeline B at R = 2), confirming that GPU-hours and opportunity cost, not electricity, are the meaningful deployment metrics at this scale.

6.3 Throughput headroom

Table 7 reports the maximum number of images each configuration could process in a 24-hour window, assuming continuous operation with no other workloads on the hardware.

Table 7. Maximum sustainable daily volume (24-hour window).

	Pipeline A GPU	Pipeline B GPU	Pipeline A CPU	Pipeline B CPU
Design daily volume (images/day)	2,215,658	293,420,539	79,681	1,313,892
Headroom factor over 4,000/day	554×	**73,355×**	20×	328×

Pipeline B GPU has throughput headroom exceeding 73,000× the design daily volume. This headroom is relevant for two near-term scenarios. First, the expected increase in WV mode acquisitions when ROSE-L (expected 2028) join the Sentinel constellation. Second, a potential expansion of the IWS to additional SAR modes beyond WV (e.g., IW and EW mode imagery), which would increase the daily volume by one to two orders of magnitude. Pipeline B GPU could absorb such increases without architectural changes; Pipeline A GPU, at 554× headroom, would require careful scheduling to accommodate a 100-fold volume increase alongside other GPU workloads.

Pipeline A CPU, with only 20× headroom, would become throughput-limited under any significant volume increase. Pipeline B CPU, at 328×, retains meaningful capacity for a GPU-free deployment at current and moderately expanded volumes.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

7. Discussion

7.1 Deployment decision

The benchmarks support a clear operational recommendation for the IWS: deploy Pipeline B (Julia Lux CNN) on GPU for all inference, with Pipeline B on CPU as a fallback for the daily feed when the GPU is unavailable.

The basis for this recommendation is threefold. First, Pipeline B GPU is the only configuration that makes archive reprocessing a routine operation. At 1.4 hours per full archive pass, a new dataset version can be issued the same day a model is validated. Pipeline A GPU requires 7.7 days per pass — tolerable if reprocessing is rare, but increasingly impractical as the retraining cadence increases toward R = 4 and as the archive continues to grow with each year of Sentinel-1 acquisitions.

Second, the GPU opportunity cost is disproportionate. Pipeline A GPU occupies the shared L40 for 384 hours per year at R = 2, displacing other institutional workloads — ocean modelling, atmospheric simulation, development — for over two weeks of cumulative wall-clock time. Pipeline B GPU occupies it for 2.9 hours. On shared infrastructure, this is the difference between a pipeline that coexists with other services and one that competes with them.

Third, Pipeline B CPU provides a viable fallback: the daily feed of 4,000 images completes in 4.4 minutes on the server's dual Xeon CPUs, with no GPU contention. This enables a hybrid operating mode where the daily feed can continue on CPU when the GPU is occupied by other services, and the GPU is engaged for the periodic archive reclassification where its throughput advantage is decisive.

The accuracy gap (1.26 percentage points F1, 0.39 percentage points AUC-ROC) is acknowledged and accepted. Both pipelines exceed 95% F1 on a balanced test set, and the IWS's expert validation loop provides a second-stage check on all positive detections. The operational value of 132-fold faster throughput, same-day archive reclassification, and minimal GPU contention outweighs the marginal accuracy improvement for this service's requirements.

7.2 Accuracy–throughput trade-off

Neither pipeline dominates the other across all dimensions. Pipeline A achieves higher classification accuracy through 305 million pretrained parameters; Pipeline B achieves higher throughput and lower energy through 283,329 purpose-built parameters. They occupy fundamentally different positions on the Pareto frontier between accuracy and computational cost.

The operationally relevant question is not which point on this frontier is abstractly better, but which the IWS requires — and the answer differs by task. For the live daily feed (4,000 images per day), throughput is unconstrained: even Pipeline A CPU at 0.92 images per second completes within 1.2 hours. Here, accuracy alone could justify Pipeline A. For archive reprocessing (17 million images, two to four times per year), throughput is the binding constraint: 7.7 days of exclusive GPU occupation per pass is operationally disruptive, while 1.4 hours is not. For the IWS, where both tasks must be served by the same infrastructure, the archive reprocessing requirement determines the deployment choice.

The F1 gap at the operational threshold is 1.26 percentage points (96.26% versus 95.00%). Whether this gap is consequential depends on the downstream use of the classifications. For the IWS's primary purpose — building a comprehensive ISW occurrence database — the expert validation loop catches a substantial fraction of false positives, and the recall-biased operating point ensures that false negatives (permanent data loss) are minimised. At 95.0% F1, Pipeline B's error rate is approximately one misclassification per 20 images, compared to one per 27 for Pipeline A. In a 4,000-image daily batch, this translates to approximately 200 versus 149 errors — a difference of 51 images per day that would require additional expert review or would be misclassified without review.

7.3 The role of transfer learning versus domain-specific training

The two pipelines represent opposite ends of a well-studied design spectrum in machine learning (Zhuang et al., 2021): Pipeline A transfers representations from a large-scale pretrained model to the target domain, while Pipeline B learns its representations entirely from domain-specific SAR imagery.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

The accuracy difference between the pipelines is not a comparison between Python and Julia as languages, nor between PyTorch and Lux as frameworks. It is predominantly a comparison between a model that leverages representations learned from 38 million diverse images (EVA02, pretrained on ImageNet-22K and merged datasets) and a model trained from scratch on the IWS's domain-specific dataset alone. The 1,076-fold difference in parameter count (305 million versus 283,329) reflects this: the transfer learning approach carries forward a vast visual vocabulary, while the from-scratch approach must build its representations entirely from SAR imagery.

This distinction matters for interpreting the trajectory of each pipeline. The Python pipeline's accuracy is anchored by the frozen EVA02 backbone: the XGBoost classifier can be retrained, but the visual representations it operates on do not change. The Julia CNN, by contrast, is retrained end-to-end with each IWS dataset expansion. As the expert-validated dataset grows — each reprocessing cycle adds reviewed classifications that feed back into the next training round — the Julia CNN has more room to improve than the EVA02 embedding space has to differentiate. Whether this trajectory will close the 1.26-percentage-point gap is an empirical question that will be answered over the next several retraining cycles.

If the task requires the kind of hierarchical visual representations that only emerge from pretraining on millions of diverse images, the gap may persist. If the task's discriminative structure is learnable from a few thousand well-curated SAR examples — as the current 95.0% F1 suggests is plausible — the gap will close as the dataset grows. Beyond dataset growth, several architectural directions could narrow the accuracy gap without sacrificing the throughput advantages of a lightweight model. Knowledge distillation — training the Julia CNN to reproduce EVA02's soft probability outputs rather than hard labels — would transfer some of the pretrained model's representational knowledge into the compact architecture at training time, with no inference cost penalty (Hinton et al., 2015). Modest architectural enrichments such as channel attention (Hu et al., 2018) or depthwise separable convolutions could increase representational capacity with a sub-linear increase in parameter count. Self-supervised pretraining on the large corpus of unlabelled WV mode imagery already ingested by the IWS — estimated at up to 17 million images — could provide a domain-specific initialisation analogous to EVA02's general-purpose pretraining, but learned entirely from SAR data. Each of these directions is compatible with the operational constraints that motivated Pipeline B: they improve training without increasing per-image inference cost. The paper does not predict which outcome is more likely; it reports the current state and identifies the trajectory.

7.4 Compiler optimisation asymmetry

The Python pipeline benefits from torch.compile (PyTorch 2.5), a graph-level JIT optimisation that fuses operations and reduces GPU kernel launch overhead. No equivalent facility exists in the Julia/Lux.jl ecosystem at the time of writing (Lux 1.9.0, Julia 1.10.5). This asymmetry favours the Python pipeline's measured throughput relative to what a future Julia toolchain might achieve.

The Julia ecosystem's compiler and GPU infrastructure are maturing rapidly. In particular, Reactant.jl (Moses et al., 2026) — which compiles Julia code through MLIR and executes it via XLA on CPU, GPU, and TPU — provides graph-level optimisation analogous to PyTorch's torch.compile, and may narrow this compilation gap in future iterations of the Julia pipeline. The throughput figures reported here for the Julia pipeline should therefore be interpreted as a current lower bound rather than an architectural ceiling.

7.5 Operational context: shared GPU infrastructure

The benchmarks were conducted on the IWS operational server, where the L40 GPU is shared across multiple services including operational ocean modelling (MOHID). The GPU-hours metric reported in Section 6 is therefore not an abstract efficiency measure but a direct constraint on institutional capacity. A pipeline that occupies the GPU for 7.7 days per archive reprocessing cycle displaces other workloads for that duration; one that completes in 1.4 hours does not.

This context also informs the value of CPU inference as a fallback. The AIR Data Centre operates four L40 GPUs shared across multiple services; even so, GPU time is a managed resource. Pipeline B CPU completes the daily 4,000-image feed in 4.4 minutes — well within the 24-hour window. When the GPU is occupied by other institutional workloads, the daily feed continues uninterrupted on CPU. The GPU is then available for archive reprocessing when scheduled, where the 132× throughput advantage of GPU over CPU makes it indispensable.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

7.6 Limitations

This study has several limitations.

First, the two pipelines differ in multiple dimensions simultaneously — model architecture, parameter count, pretraining, programming language, framework, and software dependencies. The benchmarks measure the aggregate effect of these differences as deployed; they do not isolate the contribution of any single factor. The throughput advantage reflects the combined effect of architecture size, framework, and compilation pathway — variables that are deliberately not isolated because the IWS must choose between these two complete pipelines as they exist, not between hypothetical architectures that control for individual factors. Disentangling language from model size would require building equivalent architectures in both ecosystems, which lies outside the scope of an operational deployment comparison.

Second, GPU clocks were not locked during benchmarking. Both GPU pipelines ran under identical default boost profiles, reflecting the intended operational configuration. Locking clocks to a fixed frequency would yield more reproducible power measurements but would not represent the conditions under which the IWS actually operates. The power variance across runs (coefficient of variation 5.2% for Pipeline B, 0.6% for Pipeline A) is consistent with normal boost transients and does not affect the directional comparison. Pipeline A's lower CV reflects sustained full-TDP operation where boost clocks are stable; Pipeline B's higher CV reflects brief inference bursts where the GPU transitions between idle and active states.

Third, energy is reported for the GPU pipelines only. The CPU configurations were benchmarked for throughput and accuracy to characterise the full deployment space, but their archive-reprocessing times — 12.9 days (Pipeline B) to 213 days (Pipeline A) — make them operationally noncompetitive for the IWS's primary workloads. Detailed CPU energy profiling was therefore not pursued.

Fourth, the test set (5,860 images) is the same for both pipelines and was held out from all training. However, the Python pipeline's XGBoost classifier was trained on embeddings extracted by the same EVA02 model used at inference time, while the Julia CNN was trained end-to-end from raw images. This is an inherent property of transfer learning versus from-scratch training and is not a confound — it is the comparison.

Fifth, the Python GPU pipeline used torch.compile for graph-level JIT optimisation (Section 5.1), for which no equivalent exists in the Julia/Lux.jl ecosystem at the time of writing. This asymmetry favours the Python pipeline's measured throughput; the Julia throughput should be interpreted as a current lower bound. The implication for the comparison is conservative: Julia's throughput advantage (132-fold) would be larger, not smaller, under a symmetric optimisation regime.

7.7 Generalisability

The specific performance numbers reported here are tied to the IWS's classification task, dataset, and infrastructure. The methodology — measuring per-image throughput, energy, and accuracy across deployment configurations on production hardware, then projecting to operational volumes — is directly transferable to any operational Earth observation service evaluating inference pipeline options. The pipeline architectures are not unusual: a large pretrained vision model with a lightweight classifier head versus a purpose-built domain-specific CNN is a common design choice in applied computer vision, therefore, this work can inform other EO services facing similar decisions. The structural conditions under which the IWS findings are most likely to generalise are identifiable. Services that reprocess a large and growing archive with each model iteration face the same compounding relationship between per-image cost and archive size that makes throughput decisive for the IWS. Services operating on shared GPU infrastructure, where inference competes with other workloads for accelerator time, face the same opportunity-cost calculus reported in Section 6.2. Services with a growing domain-specific labelled dataset have the same option to trade pretrained representational capacity for a lightweight, retrainable architecture — and the same expectation that the accuracy gap will narrow over time. And services where a human-in-the-loop validation stage exists downstream of the classifier can absorb a modest accuracy deficit in exchange for operational efficiency, as the IWS does.

Conversely, the trade-off may tilt differently under other conditions. If the classification task demands the kind of fine-grained visual discrimination that only emerges from large-scale pretraining — for instance, distinguishing between visually similar species in ecological monitoring — the accuracy gap may be too large to accept regardless of throughput. If the deployment environment provides dedicated GPU resources with no competing workloads, the

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

opportunity-cost argument weakens. If the archive is static and reprocessing is infrequent, the throughput advantage matters only for latency, not for sustained resource occupation. Other EO services facing pipeline selection decisions could replicate the measurement protocol described in Section 5.1 on their own workloads; the framework of analysis is general even where the specific outcome differs.

8. Conclusion

This paper compared two operationally deployed inference pipelines for automated internal solitary wave detection from Sentinel-1 WV-mode SAR imagery: Pipeline A (Python, EVA02-Large + XGBoost, 305 million parameters) and Pipeline B (Julia, Lux CNN, 283,329 parameters). Both pipelines were benchmarked on GPU and CPU on the IWS operational server at the AIR Data Centre, Terceira, Azores, using the same 5,860-image balanced test set.

Three principal findings emerge.

First, Pipeline A achieves higher classification accuracy (F1 96.26 % versus 95.00 %; AUC-ROC 99.29 % versus 98.90 %), attributable primarily to the EVA02 backbone's visual representations learned from 38 million pretraining images rather than to any language or framework difference.

Second, Pipeline B GPU is 132 times faster than Pipeline A GPU (3,396 versus 25.6 images per second) and consumes 267 times less energy per image (43.7 versus 11,690 mJ), at lower instantaneous power (153 versus 300 W). The throughput advantage translates to archive reprocessing in 1.4 hours versus 7.7 days, and annual GPU occupation of 2.9 versus 384 hours at the current reprocessing cadence.

Third, classification is bit-for-bit identical across GPU and CPU for the Julia pipeline, and consistent to within 0.02% for the Python pipeline, confirming that the deployment target can be chosen on operational grounds without accuracy trade-offs.

Based on these findings, the IWS deploys Pipeline B on GPU for all inference, with Pipeline B on CPU as a daily-feed fallback when the GPU is unavailable. The 1.26-percentage-point accuracy gap is accepted: both pipelines exceed 95% F1, the expert validation loop provides a second-stage check, and the operational benefits of same-day archive reclassification and minimal GPU contention on shared infrastructure outweigh the marginal accuracy difference.

The methodology presented here — benchmarking per-image throughput, energy, and accuracy on production hardware, then projecting to operational volumes — is directly transferable to any operational Earth observation service evaluating inference pipeline options. The specific numbers are tied to the IWS; the approach is general.

Looking forward, the Julia CNN's accuracy trajectory is steeper than the Python pipeline's: as the IWS expert-validated dataset grows with each retraining cycle, the domain-specific CNN has more room to close the accuracy gap than the frozen EVA02 embedding space has to widen it. Developments in the Julia GPU toolchain — particularly graph-compilation backends such as Reactant.jl — may further improve Pipeline B's throughput. Future work will also evaluate both pipelines on additional Sentinel-1 acquisition modes beyond WV, including IW and EW, as well as emerging SAR sources which will substantially increase the daily processing volume.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

Code and Data Availability

The code for both pipelines is available from the project repository at https://github.com/aircentre/iws-pipeline-comparison-gmd under the MIT licence. The exact version used to produce the results in this paper, including benchmark scripts, output data, trained model weights, and plotting scripts, is archived on Zenodo at https://doi.org/10.5281/zenodo.19322369 (Pinelo et al., 2026). The evaluation dataset (5,860 Sentinel-1 WV mode SAR vignettes; approx. 11GB) is additionally available at https://s3.ac-az1.aircentre.org/public/iws/paper_gmd_2026-03/dataset.tar.gz (last access: 30 March 2026).

Author Contributions

Author contributions. JP conceived the study, designed the benchmarking methodology and supervised its execution, performed the formal analysis and data validation (auditing all benchmark outputs and verifying model architecture consistency across all four pipeline configurations), revised and restructured the benchmarking scripts and repository for publication, created Figure 1, and wrote the original draft. AS developed the SAR_CNN v2 model architecture in Julia and executed the benchmarks on the operational server. GT designed and implemented the Python EVA02+XGBoost pipeline. ASF curated the expert-validated ISW dataset and reviewed Sections 1 and 2. JG deployed both pipelines in containerised environments, developed the IWS platform, and maintains the operational infrastructure. JM developed the data harvesting system in Julia. All authors reviewed and approved the final manuscript.

Competing Interests Declaration

The authors declare that they have no conflict of interest. Gilberto Titericz is affiliated with NVIDIA Inc. and was never involved in any benchmark planning or execution. NVIDIA Inc. has not lent or given any hardware to the AIR Centre at any point.

Financial Support

No specific grant from any funding agency was received for this work.

Acknowledgements

The authors acknowledge the AIR Centre for institutional support. The authors used an AI language model (Claude, Anthropic) to assist with some sentence-level English-language editing and to format repository documentation. All scientific content, analysis, and interpretation are the authors' own.

References

Alford, M. H., Peacock, T., MacKinnon, J. A., Nash, J. D., Buijsman, M. C., Centurioni, L. R., Chao, S.-Y., Chang, M.-H., Farmer, D. M., Fringer, O. B., Fu, K.-H., Gallacher, P. C., Graber, H. C., Helfrich, K. R., Jachec, S. M., Jackson, C. R., Klymak, J. M., Ko, D. S., Jan, S., Johnston, T. M. S., Legg, S., Lee, I.-H., Lien, R.-C., Mercier, M. J., Moum, J. N., Musgrave, R., Park, J.-H., Pickering, A. I., Pinkel, R., Rainville, L., Ramp, S. R., Rudnick, D. L., Sarkar, S., Scotti, A., Simmons, H. L., St Laurent, L. C., Venayagamoorthy, S. K., Wang, Y.-H., Wang, J., Yang, Y. J., Paluszkiewicz, T., and (David) Tang, T.-Y.: The formation and fate of internal waves in the South China Sea, Nature, 521, 65–69, https://doi.org/10.1038/nature14399, 2015.

Bao, S., Meng, J., Sun, L., and Liu, Y.: Detection of ocean internal waves based on Faster R-CNN in SAR images, J. Oceanol. Limnol., 38, 55–63, https://doi.org/10.1007/s00343-019-9028-6, 2020.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

Bezanson, J., Edelman, A., Karpinski, S., and Shah, V. B.: Julia: A Fresh Approach to Numerical Computing, SIAM Review, 59, 65–98, https://doi.org/10.1137/141000671, 2017.

Chang, M. H., Lien, R. C., Lamb, K. G., and Diamessis, P. J.: Long-Term Observations of Shoaling Internal Solitary Waves in the Northern South China Sea, J. Geophys. Res. Oceans, 126, https://doi.org/10.1029/2020JC017129, 2021.

Chen, T. and Guestrin, C.: XGBoost: A scalable tree boosting system, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785–794, https://doi.org/10.1145/2939672.2939785, 2016.

Moses, W. S., Pal, A., and Reactant.jl contributors: Reactant.jl: Optimizing Julia functions with MLIR and XLA for high-performance execution on CPU, GPU, and TPU, https://github.com/EnzymeAD/Reactant.jl, last access: 24 March 2026.

Fang, Y., Sun, Q., Wang, X., Huang, T., Wang, X., and Cao, Y.: EVA-02: A Visual Representation for Neon Genesis, Image Vision Comput., 149, 105171, https://doi.org/10.1016/j.imavis.2024.105171, 2023.

Garrett, C. and Munk, W.: Internal Waves in the Ocean, Annu. Rev. Fluid Mech., 11, 339–369, https://doi.org/10.1146/annurev.fl.11.010179.002011, 1979.

Helfrich, K. R. and Melville, W. K.: Long Nonlinear Internal Waves, Annu. Rev. Fluid Mech., 38, 395–425, https://doi.org/10.1146/annurev.fluid.38.050304.092129, 2006.

Hinton, G., Vinyals, O., and Dean, J.: Distilling the Knowledge in a Neural Network, arXiv preprint arXiv:1503.02531, 2015.

Hu, J., Shen, L., and Sun, G.: Squeeze-and-Excitation Networks, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 7132–7141, https://doi.org/10.1109/CVPR.2018.00745, 2018.

Jackson, C. R., da Silva, J. C. B., and Jeans, G.: The Generation of Nonlinear Internal Waves, Oceanography, 25, 108–123, https://doi.org/10.5670/oceanog.2012.46, 2012.

Magalhaes, J. M., Da Silva, J. C. B., Buijsman, M. C., and Garcia, C. A. E.: Effect of the North Equatorial Counter Current on the generation and propagation of internal solitary waves off the Amazon shelf (SAR observations), Ocean Science, 12, 243–255, https://doi.org/10.5194/os-12-243-2016, 2016.

Osborne, A. R. and Burch, T. L.: Internal Solitons in the Andaman Sea, Science, 208, 451–460, https://doi.org/10.1126/science.208.4443.451, 1980.

Pal, A.: On Efficient Training and Inference of Neural Differential Equations, Master of Science, Massachusetts Institute of Technology, 2023.

Pal, A.: Lux: Explicit Parameterization of Deep Neural Networks in Julia, https://doi.org/10.5281/ZENODO.7808904, 2026.

Pinelo, J., Shukla, A., Titericz, G., Santos-Ferreira, A., Gonçalves, J., and Moniz, J.: Code and data for: Choosing an operational inference pipeline for internal solitary wave detection in Sentinel-1 SAR imagery: EVA02+XGBoost versus a Julia Lux CNN (GMD, 2026), Zenodo [code], https://doi.org/10.5281/zenodo.19322369, 2026.

Pinelo, J., Santos-Ferreira, A. M., Gonçalves, J., Titericz, G., da Silva, J. C. B., Johannessen, J. A., and Chapron, B.: IWS - Internal Waves Service: a world-first repository for planetary-scale internal solitary waves monitoring, in: Proc. SPIE, https://doi.org/10.1117/12.3069138, 2025.

Roustan, J. B., Bordois, L., Garciá-Lafuente, J., Dumas, F., Auclair, F., and Carton, X.: Evidence of Reflected Internal Solitary Waves in the Strait of Gibraltar, J. Geophys. Res. Oceans, 129, https://doi.org/10.1029/2023JC020152, 2024.

https://doi.org/10.5194/egusphere-2026-1798
Preprint. Discussion started: 17 June 2026

735 Santos-Ferreira, A. M., Pinelo, J., da Silva, J. C. B., Johannessen, J. A., Chapron, B., Gommenginger, C.,
736 Magalhães, J. M., Buijsman, M., Forget, G., Pineda, J., Goh, E., Ávila, M., Diogo, I., and Gonçalves, J.: The Internal
737 Waves Service Workshop: Observing Internal Waves Globally with Deep Learning and Synthetic Aperture Radar,
738 Bull. Am. Meteorol. Soc., 106, E1462–E1470, https://doi.org/10.1175/bams-d-25-0133.1, 2025.

739 Schwartz, R., Dodge, J., Smith, N. A., and Etzioni, O.: Green AI, Commun. ACM, 63, 54–63,
740 https://doi.org/10.1145/3381831, 2020.

741 da Silva, J. C. B., Buijsman, M. C., and Magalhaes, J. M.: Internal waves on the upstream side of a large sill of the
742 Mascarene Ridge: A comprehensive view of their generation mechanisms and evolution, Deep. Sea. Res. 1.
743 Oceanogr. Res. Pap., 99, 87–104, https://doi.org/10.1016/j.dsr.2015.01.002, 2015.

744 Torres, R., Snoeij, P., Geudtner, D., Bibby, D., Davidson, M., Attema, E., Potin, P., Rommen, B., Floury, N.,
745 Brown, M., Traver, I. N., Deghaye, P., Duesmann, B., Rosich, B., Miranda, N., Bruno, C., L'Abbate, M., Croci, R.,
746 Pietropaolo, A., Huchler, M., and Rostan, F.: GMES Sentinel-1 mission, Remote Sens. Environ., 120, 9–24,
747 https://doi.org/10.1016/j.rse.2011.05.028, 2012.

748 Whalen, C. B., de Lavergne, C., Naveira Garabato, A. C., Klymak, J. M., MacKinnon, J. A., and Sheen, K. L.:
749 Internal wave-driven mixing: governing processes and consequences for climate, Nat. Rev. Earth Environ., 1, 606–
750 621, https://doi.org/10.1038/s43017-020-0097-z, 2020.

751 Wightman, R., Raw, N., Soare, A., Arora, A., Ha, C., Reich, C., Guan, F., Kaczmarzyk, J., Rizin, M., Kim, H.,
752 Kertész, C., Mehta, D., Cucurull, G., Singh, K., hankyul, Tatsunami, Y., Lavin, A., Zhuang, J., Hollemans, M.,
753 Rashad, M., Sameni, S., Shults, V., Lucain, Wang, X., Kwon, Y., and Uchida, Y.: rwightman/pytorch-image-
754 models: v0.8.10dev0 Release, https://doi.org/10.5281/ZENODO.7618837, 2023.

755 Zhang, S., Li, X., and Zhang, X.: Internal Wave Signature Extraction From SAR and Optical Satellite Imagery
756 Based on Deep Learning, IEEE Transactions on Geoscience and Remote Sensing, 61, 1–16,
757 https://doi.org/10.1109/TGRS.2023.3258189, 2023.

758 Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q.: A Comprehensive Survey on Transfer
759 Learning, Proceedings of the IEEE, 109, 43–76, https://doi.org/10.1109/JPROC.2020.3004555, 2021.