

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA E ELETRÔNICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELETRÔNICA

Libni da Silva Junior

**Classificação Embarcada de Imagens de
Observação da Terra em Microcontrolador de
Baixo Custo para Apoio à Decisão de Downlink
em Nanosatélites**

Florianópolis

2026

Libni da Silva Junior

Classificação Embarcada de Imagens de Observação da Terra em Microcontrolador de Baixo Custo para Apoio à Decisão de Downlink em Nanosatélites

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Engenharia Eletrônica da Universidade Federal de Santa Catarina como requisito parcial para a obtenção do título de Bacharel em Engenharia Eletrônica.

Universidade Federal de Santa Catarina

Orientador: Prof. Dr. Eduardo Bezerra

Florianópolis
2026

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Junior, Libni da Silva
Classificação Embarcada de Imagens de Observação da Terra
em Microcontrolador de Baixo Custo para Apoio à Decisão de
Downlink em Nanosatélites / Libni da Silva Junior ;
orientador, Eduardo Augusto Bezerra, 2026.
77 p.

Trabalho de Conclusão de Curso (graduação) -
Universidade Federal de Santa Catarina, Centro Tecnológico,
Graduação em Engenharia Eletrônica, Florianópolis, 2026.

Inclui referências.

1. Engenharia Eletrônica. 2. TinyML. 3. Observação da
Terra. 4. Redes neurais convolucionais. 5. Nanosatélites.
I. Bezerra, Eduardo Augusto. II. Universidade Federal de
Santa Catarina. Graduação em Engenharia Eletrônica. III.
Titulo.

Libni da Silva Junior

Classificação Embarcada de Imagens de Observação da Terra em Microcontrolador de Baixo Custo para Apoio à Decisão de Downlink em Nanosatélites

Este Trabalho de Conclusão de Curso foi julgado adequado para a obtenção do título de Bacharel em Engenharia Eletrônica e aprovado em sua forma final pelo Curso de Graduação em Engenharia Eletrônica.

Florianópolis, 2026



Documento assinado digitalmente
Fernando Rangel de Sousa
Data: 07/07/2026 14:29:10-0300
CPF: ***.649.114-**
Verifique as assinaturas em <https://v.ufsc.br>

Coordenação do Curso

Banca Examinadora



Documento assinado digitalmente
Eduardo Augusto Bezerra
Data: 07/07/2026 11:18:23-0300
CPF: ***.851.577-**
Verifique as assinaturas em <https://v.ufsc.br>

Eduardo Augusto Bezerra, Ph.D.
Orientador
UFSC



Documento assinado digitalmente
Joao Claudio Elsen Barcellos
Data: 07/07/2026 19:00:30-0300
CPF: ***.320.029-**
Verifique as assinaturas em <https://v.ufsc.br>

Eng. João Cláudio Elsen Barcellos
Avaliador
UFSC



Documento assinado digitalmente
JOCELI MAYER
Data: 07/07/2026 16:23:40-0300
CPF: ***.833.519-**
Verifique as assinaturas em <https://v.ufsc.br>

Joceli Mayer, Ph.D.
Avaliador
UFSC

RESUMO

Nanosatélites de observação da Terra geram volumes de imagens que superam a capacidade de transmissão disponível em links de downlink UHF, tornando a triagem automática de dados a bordo uma necessidade operacional. Este trabalho propõe, implementa e valida um sistema de classificação embarcada de imagens de observação da Terra executável em microcontrolador de baixo custo, com o objetivo de apoiar a decisão automática de transmitir ou descartar imagens antes do downlink. O sistema é construído sobre o microcontrolador de baixo custo ESP32-S3-WROOM-1-N16R8, equipado com memória PSRAM expandida, rodando o modelo *cnn_v2_64* quantizado em INT8 via TensorFlow Lite for Microcontrollers. O modelo classifica imagens de 64×64 pixels RGB em quatro classes de cena — *terra*, *oceano*, *nuvem* e *mista* — treinadas a partir de dados públicos EuroSAT e CloudSEN12. A avaliação em hardware real sobre 492 amostras do conjunto de teste observou acurácia balanceada de 82,1% com zero degradação pós-quantização INT8, latência de inferência de 6,1 s (arena TFLM de 551 KB em PSRAM) e jitter de 1 ms em 492 execuções consecutivas, indicando comportamento determinístico. A política de decisão S3 *Dual Threshold* (descartar somente se $\text{pred} = \text{nuvem} \wedge \text{conf} \geq 0,40 \wedge P(\text{nuvem}) \geq 0,40$) é fundamentada em teoria de decisão custo-assimétrica e alcança Recall de 90,5% com 21,7% de imagens descartadas. A análise energética estima 8 J por inferência ($\pm 30\%$) e distingue o descarte total observado (21,7%) do subconjunto conservador de nuvens corretamente filtradas (14,6%), indicando saldo positivo a 4.800 bps e resultado dependente dessa leitura a 9.600 bps. O sistema é validado de forma desacoplada do hardware de imageamento real (SLCam/SpaceLab/UFSC) por meio de protocolo serial de validação em laboratório, com adaptador *host* em Python processando 492 amostras sem erros. Os resultados demonstram a viabilidade da abordagem em contexto de prova de conceito, utilizando imagens de datasets públicos como representação do domínio de observação da Terra.

Palavras-chave: CubeSat. Classificação de imagens. Earth Observation. ESP32-S3. Nanosatélite. TinyML.

ABSTRACT

Earth observation nanosatellites generate image volumes that exceed the downlink capacity of typical UHF links, making onboard automatic data triage an operational necessity. This work proposes, implements, and validates an embedded image classification system executable on a low-cost microcontroller, aimed at supporting automatic transmit-or-discard decisions prior to downlink. The system runs on a low-cost ESP32-S3-WROOM-1-N16R8 microcontroller with expanded PSRAM, executing the *cnn_v2_64* model quantized to INT8 via TensorFlow Lite for Microcontrollers. The model classifies 64×64 RGB images into four scene classes — *land*, *ocean*, *cloud*, and *mixed* — trained on public EuroSAT and CloudSEN12 datasets. Evaluation on real hardware over 492 test samples observed 82.1% balanced accuracy with zero post-quantization degradation, 6.1 s inference latency (551 KB TFLM arena in PSRAM), and 1 ms jitter across 492 consecutive runs, indicating deterministic behavior. The S3 *Dual Threshold* decision policy (discard only if $\text{pred} = \text{cloud} \wedge \text{conf} \geq 0.40 \wedge \text{P}(\text{cloud}) \geq 0.40$) is grounded in cost-sensitive decision theory and achieves 90.5% Recall while discarding 21.7% of images. Energy analysis estimates 8 J per inference ($\pm 30\%$) and distinguishes total observed discard (21.7%) from the conservative subset of correctly filtered cloud images (14.6%), indicating positive balance at 4,800 bps and an outcome that depends on this interpretation at 9,600 bps. The system is validated decoupled from the actual imaging hardware (SLCam/SpaceLab/UFSC) through a serial validation protocol in laboratory, with a Python host adapter processing 492 samples without errors. Results demonstrate the feasibility of the approach as a proof of concept using public benchmark datasets as a proxy for the Earth observation domain.

Keywords: CubeSat. Earth Observation. Embedded image classification. ESP32-S3. Nanosatellite. TinyML.

LISTA DE ILUSTRAÇÕES

Figura 1 – Fluxo de triagem embarcada de imagens de observação da Terra	17
Figura 2 – Pipeline experimental em quatro estágios com artefatos de rastreabilidade	38
Figura 3 – Pipeline de composição do benchmark curado a partir de duas fontes públicas	39
Figura 4 – Arquitetura do modelo <i>cnn_v2_64</i> — blocos convolucionais e dimensões	41
Figura 5 – Fluxograma da política S3 <i>Dual Threshold</i>	49
Figura 6 – Curva de treinamento do <i>cnn_v2_64</i> — run definitiva (71 épocas) . .	51
Figura 7 – Heatmap da matriz de confusão — <i>cnn_v2_64</i> em hardware INT8 ($n = 492$)	52
Figura 8 – Acurácia balanceada por arquitetura — <i>test set</i> ($n = 492$), hardware INT8	53
Figura 9 – Distribuição de memória no ESP32-S3-WROOM-1 durante inferência (alturas ilustrativas)	54
Figura 10 – Saldo energético líquido por imagem em função da taxa de descarte . .	57

LISTA DE TABELAS

Tabela 1 – Evolução histórica do processamento embarcado em missões de imageamento	22
Tabela 2 – Remapeamento das classes EuroSAT para as classes da missão	28
Tabela 3 – Composição do benchmark curado (benchmark-v1.csv)	29
Tabela 4 – Comparativo de sistemas de classificação embarcada em nanosatélites .	30
Tabela 5 – Matriz de custo para a decisão TX/DX em missão orbital	31
Tabela 6 – Energia de transmissão por imagem (64×64 RGB) em função da taxa de downlink	35
Tabela 7 – Análise de ponto de equilíbrio energético por taxa de downlink	36
Tabela 8 – Composição do benchmark benchmark-v1.csv	40
Tabela 9 – Hiperparâmetros de treinamento (protocolo v2)	42
Tabela 10 – Comparativo de modelos — acurácia e viabilidade de embarcamento .	43
Tabela 11 – Comparativo das estratégias de decisão simuladas (492 amostras) . . .	48
Tabela 12 – Comparativo de modelos — conjunto fixo de comparação ($n = 492$, $seed = 42$)	50
Tabela 13 – Desempenho por classe — <i>cnn_v2_64</i> , hardware real ($n = 492$)	51
Tabela 14 – Matriz de confusão — <i>cnn_v2_64</i> em hardware INT8 ($n = 492$)	51
Tabela 15 – Latência de inferência — ESP32-S3, run 20260520_151533 ($n = 492$) .	54
Tabela 16 – Comparativo de estratégias de decisão — hardware real ($n = 492$) . . .	55
Tabela 17 – Comparativo com sistemas análogos de inferência embarcada em nanosatélites	58
Tabela 18 – Hiperparâmetros de treinamento do modelo <i>cnn_v2_64</i>	68
Tabela 19 – Camadas do modelo <i>cnn_v2_64</i> (inferência, training=False)	69
Tabela 20 – Pegada de memória do <i>cnn_v2_64</i> em diferentes formatos	70
Tabela 21 – Configuração da interface UART do ESP32-S3	71
Tabela 22 – Comandos do protocolo UART	71
Tabela 23 – Métricas de todos os modelos no conjunto de teste ($n = 492$)	73
Tabela 24 – Comparação float32 vs. INT8 para modelos candidatos ($n = 492$) . . .	73
Tabela 25 – Intervalos de confiança de 95% para <i>bal_acc</i> ($n = 492$)	74
Tabela 26 – Melhores variantes de cada estratégia de decisão (simulação float32, $n = 492$)	74
Tabela 27 – Identificação e propriedades gerais do <i>benchmark-v1</i>	75
Tabela 28 – Distribuição de amostras por fonte e classe no <i>benchmark-v1</i>	75
Tabela 29 – Distribuição de amostras por <i>split</i> e classe no <i>benchmark-v1</i>	76

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
bal_acc	Acurácia Balanceada (<i>Balanced Accuracy</i>)
BN	Normalização em Lote (<i>Batch Normalization</i>)
CNN	Rede Neural Convolucional (<i>Convolutional Neural Network</i>)
CubeSat	Nanosatélite padrão de 10 cm × 10 cm × 10 cm por unidade (1U)
DX	Decisão de descarte (<i>Discard</i>)
EEL	Departamento de Engenharia Elétrica e Eletrônica (UFSC)
EO	Observação da Terra (<i>Earth Observation</i>)
ESP32-S3	Microcontrolador Espressif ESP32-S3-WROOM-1-N16R8
FN	Falso Negativo (<i>False Negative</i>)
FP	Falso Positivo (<i>False Positive</i>)
FPS	Quadros por segundo (<i>Frames Per Second</i>)
IC	Intervalo de Confiança
INT8	Quantização em inteiro de 8 bits
IoT	Internet das Coisas (<i>Internet of Things</i>)
LEO	Órbita Terrestre Baixa (<i>Low Earth Orbit</i>)
LULC	Uso e Cobertura da Terra (<i>Land Use and Land Cover</i>)
MAC	Operação Multiplicação-Acumulação (<i>Multiply-Accumulate</i>)
MCU	Microcontrolador (<i>Microcontroller Unit</i>)
NAS	Busca de Arquitetura Neural (<i>Neural Architecture Search</i>)
NPU	Unidade de Processamento Neural (<i>Neural Processing Unit</i>)
PSRAM	Memória RAM Pseudo-Estática (<i>Pseudo-Static RAM</i>)
PTQ	Quantização Pós-Treinamento (<i>Post-Training Quantization</i>)
QAT	Quantização com Treinamento (<i>Quantization-Aware Training</i>)

SIMD	Instrução Única, Múltiplos Dados (<i>Single Instruction, Multiple Data</i>)
SLCam	SpaceLab Camera (UFSC)
SRAM	Memória RAM Estática (<i>Static RAM</i>)
TFLM	TensorFlow Lite for Microcontrollers
TinyML	Aprendizado de Máquina em dispositivos com recursos restritos
TN	Verdadeiro Negativo (<i>True Negative</i>)
TP	Verdadeiro Positivo (<i>True Positive</i>)
TX	Decisão de transmissão (<i>Transmit</i>)
UHF	Ultra Alta Frequência (<i>Ultra High Frequency</i>)
UFSC	Universidade Federal de Santa Catarina
UART	Receptor/Transmissor Assíncrono Universal (<i>Universal Asynchronous Receiver/Transmitter</i>)

LISTA DE SÍMBOLOS

$\hat{p}$	Proporção amostral estimada
n	Tamanho amostral
S	Fator de escala na quantização INT8
Z	Ponto zero na quantização INT8
q	Valor quantizado (inteiro INT8)
r	Valor real (float32) correspondente
E	Energia (J)
P	Potência (W)
t	Tempo (s)
R^*	Taxa de descarte de equilíbrio (<i>break-even</i>)
θ^*	Limiar ótimo de decisão custo-assimétrica

SUMÁRIO

1	INTRODUÇÃO	14
1.1	Contexto e Motivação	14
1.2	Definição do Problema	16
1.3	Objetivos	17
1.3.1	Objetivo Geral	17
1.3.2	Objetivos Específicos	17
1.4	Contribuições	18
1.5	Delimitação do Escopo	19
1.6	Organização do Trabalho	20
2	FUNDAMENTAÇÃO TEÓRICA	21
2.1	Triagem Embarcada em Missões de Imageamento	21
2.1.1	Do Processamento por Regras ao Aprendizado de Máquina	21
2.1.2	Classificação de Cena como Tarefa de Triagem	22
2.2	TinyML em Microcontroladores Embarcados	22
2.2.1	Restrições de Memória: Pesos vs. Arena de Ativações	22
2.2.2	Benchmarking e Protocolos de Avaliação	23
2.2.3	Estratégias de Otimização para MCUs	23
2.2.4	O ESP32-S3 no Ecossistema TinyML	24
2.3	Quantização de Redes Neurais	24
2.3.1	Fundamentos: Quantização Afim	25
2.3.2	Quantização Pós-Treinamento (PTQ)	25
2.3.3	Resultados de Quantização no Presente Trabalho	26
2.4	Classificação de Cena em Imagens de Satélite	26
2.4.1	Caracterização da Tarefa	26
2.4.2	Dataset EuroSAT	27
2.4.3	Dataset CloudSEN12	28
2.4.4	Benchmark Curado: Composição Final	28
2.5	Trabalhos Análogos em Nanosatélites	29
2.6	Teoria de Decisão Custo-Assimétrica	30
2.6.1	Framework de Decisão Ótima sob Custos Assimétricos	30
2.6.2	Matriz de Custo para a Missão Orbital	31
2.6.3	Limiar Ótimo de Decisão	31
2.6.4	MScore: Métrica de Avaliação de Política	32
2.7	Métricas de Avaliação para Sistemas Embarcados	32

2.7.1	Acurácia Balanceada	32
2.7.2	Intervalo de Confiança para Acurácia	33
2.7.3	Latência de Inferência	33
2.7.4	Arena TFLM	33
2.7.5	Energia por Inferência	34
2.8	Análise Energética: Inferência versus Transmissão	34
2.8.1	Energia de Transmissão por Imagem	34
2.8.2	Energia de Inferência (Estimada)	34
2.8.3	Ponto de Equilíbrio Energético	35
2.8.4	Interpretação dos Resultados Energéticos	36
3	METODOLOGIA	37
3.1	Visão Geral do Pipeline	37
3.2	Dataset Composto	38
3.2.1	Estratégia Multi-Fonte	38
3.2.2	Pré-processamento e Remapeamento	38
3.2.2.0.1	Remapeamento das classes.	39
3.2.2.0.2	Padronização de resolução.	39
3.2.3	Composição do Benchmark e Divisão dos Conjuntos	39
3.3	Arquiteturas Avaliadas	40
3.3.1	CNN Custom (cnn_v2)	40
3.3.2	Modelos de Transfer Learning	41
3.3.3	Baseline Clássica	41
3.4	Protocolo de Treinamento	42
3.4.1	Configuração Comum	42
3.4.2	Aumento de Dados	42
3.4.3	Curva de Treinamento e Early Stopping	43
3.5	Seleção de Modelo	43
3.6	Exportação e Quantização TFLite INT8	44
3.7	Implementação no ESP32-S3	45
3.7.1	Hardware	45
3.7.2	Pilha de Software	45
3.7.3	Protocolo de Comunicação UART	46
3.8	Protocolo de Avaliação em Hardware	46
3.9	Política de Decisão de Missão	47
3.9.1	Derivação da Política	47
3.9.2	Política Adotada: S3 <i>Dual Threshold</i>	48
3.9.3	Validação da Política em Hardware INT8	49
4	RESULTADOS E DISCUSSÃO	50

4.1	Avaliação Offline dos Modelos	50
4.1.1	Comparativo de Acurácia	50
4.1.2	Análise por Classe	50
4.1.3	Matriz de Confusão e Análise de Erros	51
4.2	Quantização INT8 — Zero Degradação	52
4.3	Caracterização de Hardware (ESP32-S3)	53
4.3.1	Arena TFLM e Memória	53
4.3.2	Latência e Determinismo	53
4.4	Política de Decisão de Missão	54
4.4.1	Comparativo de Estratégias	54
4.4.2	Equivalência S3 vs. S0 em Hardware INT8	54
4.5	Análise Energética	55
4.5.1	Energia de Transmissão	55
4.5.2	Energia de Inferência (Estimada)	56
4.5.3	Análise de Break-even	56
4.5.4	Valor Operacional Além da Energia	56
4.6	Posicionamento em Relação ao Estado da Arte	57
5	CONCLUSÃO	59
5.1	Síntese dos Resultados	59
5.2	Limitações	59
5.3	Pré-requisitos para Implantação Operacional	61
5.4	Trabalhos Futuros	62
5.5	Considerações Finais	63
	Referências	64
	APÊNDICES	67
	APÊNDICE A – ARQUITETURA DO MODELO <i>CNN_V2_64</i>	68
	APÊNDICE B – PROTOCOLO UART COMPLETO	71
	APÊNDICE C – RESULTADOS DETALHADOS DO BENCHMARK	73
	APÊNDICE D – MANIFESTO DO DATASET BENCHMARK-V1	75

1 INTRODUÇÃO

1.1 CONTEXTO E MOTIVAÇÃO

A observação da Terra a partir de satélites de pequeno porte experimenta crescimento acelerado impulsionado pela miniaturização de sensores ópticos, pela padronização da plataforma CubeSat e pela redução dos custos de lançamento (Lasseur *et al.*, 2024). Uma unidade padrão 1U (10 cm × 10 cm × 10 cm) com câmera de resolução espacial de até 100 m é hoje operável por grupos universitários, como o Laboratório Espacial da Universidade Federal de Santa Catarina (SpaceLab/UFSC), que desenvolve o sistema de imageamento SLCam (*SpaceLab Camera*) para futura missão orbital.

Missões de imageamento em CubeSats operam sob três restrições acopladas: orçamento de energia (geração solar limitada a poucos watts), capacidade de armazenamento (Flash de dezenas a centenas de megabytes) e **capacidade de retorno de dados ao solo** (downlink). Embora energia e armazenamento sejam limitantes em outros contextos operacionais, o **downlink é o gargalo dominante para o retorno de dados em missões de imageamento contínuo**: um CubeSat em órbita terrestre baixa (LEO) dispõe de janelas de visibilidade com estações terrestres de apenas 5 a 15 minutos por passagem orbital, com taxa de transmissão em UHF tipicamente entre 1.200 e 9.600 bps (Saeed *et al.*, 2019). Saeed et al. (Saeed *et al.*, 2019) identificam o orçamento de downlink como a limitação operacional dominante para missões de imageamento, independentemente de quanto dado pode ser gerado ou armazenado a bordo.

A discrepância entre capacidade de captura e de transmissão é drástica e pode ser quantificada diretamente. Uma imagem de 64×64 pixels RGB sem compressão ocupa:

$$N_{\text{bits}} = 64 \times 64 \times 3 \times 8 = 98.304 \text{ bits} \approx 98,3 \text{ kbits} \quad (1.1)$$

A uma taxa UHF de 4.800 bps (configuração típica de CubeSats universitários como o BIRDS-3 (Mehrpour *et al.*, 2017)), o tempo de transmissão por imagem é:

$$t_{\text{TX}} = \frac{98.304 \text{ bits}}{4.800 \text{ bps}} \approx 20,5 \text{ s/imagem} \quad (1.2)$$

Em uma janela de visibilidade de 10 minutos (600 s), o satélite pode transmitir no máximo:

$$N_{\text{TX}} = \left\lfloor \frac{600}{20,5} \right\rfloor = 29 \text{ imagens por passagem} \quad (1.3)$$

No mesmo intervalo de 10 minutos, uma câmera com taxa de captura de apenas 1 FPS poderia registrar 600 imagens. Isso representa uma **razão de compressão neces-**

sária de aproximadamente 20:1: mais de 95% das imagens capturáveis não cabem no orçamento de downlink por passagem.

O problema é agravado pela prevalência de cobertura de nuvens: estudos de climatologia de satélite indicam que mais de 60% das imagens capturadas em órbita apresentam cobertura de nuvens suficiente para obscurecer completamente a superfície (Giuffrida *et al.*, 2020). Sem triagem prévia, das 29 imagens transmitidas por passagem, apenas cerca de 12 conteriam dado de superfície cientificamente útil; as outras 17 (58% da banda) seriam ocupadas por imagens de nuvem descartáveis após a recepção. Com triagem automática embarcada, todas as 29 vagas da janela de downlink poderiam ser preenchidas com imagens úteis, representando um **ganho de eficiência de até 2,4×** sem qualquer alteração de hardware de comunicação.

A solução estrutural é a **triagem automática embarcada**: classificar cada imagem a bordo do satélite imediatamente após a captura e descartar, antes do downlink, aquelas sem interesse para a missão. Essa estratégia é estabelecida na literatura desde 2004, quando Tran *et al.* (Tran *et al.*, 2004) demonstraram, no experimento ASE (*Autonomous Sciencecraft Experiment*) a bordo do satélite EO-1 da NASA, reduções superiores a 70% no volume de dados transmitidos sem perda de dado científico relevante, estabelecendo o paradigma de processamento embarcado que motiva o presente trabalho.

Missões subsequentes indicaram a viabilidade operacional dessa abordagem com redes neurais. Em 2020, CloudScout, embarcado no satélite PhiSat-1 da Agência Espacial Europeia (ESA), executou inferência de rede neural convolucional em um processador Intel Myriad 2 para filtrar imagens completamente nubladas antes da transmissão (Giuffrida *et al.*, 2020). No mesmo ano, o grupo do Instituto Kyushu de Tecnologia implantou o CubeSatNet, uma CNN ultraleve em microcontrolador ARM Cortex a bordo de um CubeSat 1U real (BIRDS-3), demonstrando 90% de acurácia na classificação binária de imagens, com redução potencial de dois terços do tempo de downlink (Maskey; Cho, 2020). Em 2023, WorldFloods foi implantado no satélite Phi-sat-2 (ESA), executando segmentação semântica de inundações diretamente a bordo e transmitindo apenas mapas de decisão em vez de imagens brutas (Mateo-Garcia *et al.*, 2023). Chatar *et al.* (Chatar *et al.*, 2024) demonstraram, em 2024, priorização de downlink por classificação embarcada em um CubeSat 6U (missão VERTECS), indicando que a abordagem se generaliza além da detecção de nuvens.

A revisão sistemática de Lasseur *et al.* (Lasseur *et al.*, 2024) cataloga o estado atual de hardware e software para inferência neural embarcada em satélites e identifica microcontroladores comerciais (*COTS*) como alternativa de menor custo para nanosatélites com orçamento restrito. O trabalho contemporâneo mais próximo ao presente TCC é o de Capogrosso e Magno (Capogrosso; Magno, 2026), que reportam, em paper aceito para publicação em 2026, latências de 3 a 30 ms e energias de 0,68 a 6,45 mJ por inferência em um STM32N6 (ARM Cortex-M55 com NPU dedicada) para classificação de imagens

de observação da Terra. A diferença de três ordens de grandeza frente às estimativas do presente trabalho (8 J) decorre diretamente da ausência de NPU no ESP32-S3 e do uso de PSRAM em vez de SRAM interna, conforme discutido no Capítulo 4.

Dentro desse contexto, o SpaceLab/UFSC desenvolve o SLCam: um payload de câmera para nanosatélites projetado para capturar imagens da Terra em missões CubeSat universitárias. A integração de um módulo de classificação inteligente ao SLCam permitiria a triagem automática antes do downlink, ampliando a autonomia operacional e recuperando a banda atualmente desperdiçada com imagens de nuvem. O presente trabalho investiga a viabilidade técnica desse módulo de classificação utilizando o ESP32-S3, um *System-on-Chip* comercial sem NPU dedicada e com custo inferior a USD 10, como plataforma de inferência, tornando a solução acessível para CubeSats universitários.

1.2 DEFINIÇÃO DO PROBLEMA

O problema central deste trabalho é: *é possível classificar imagens de observação da Terra em classes de cena semanticamente úteis diretamente a bordo de um microcontrolador ESP32-S3, sem acelerador de IA dedicado, com acurácia e latência suficientes para apoiar uma decisão automática e conservadora de transmitir ou descartar cada imagem, dentro das restrições de memória e energia da plataforma?*

A decisão de transmitir ou descartar é inerentemente assimétrica: descartar erroneamente uma imagem útil (*falso negativo*) representa uma perda irreversível (o satélite revisita a mesma área apenas dias depois), enquanto transmitir uma imagem de nuvem inútil (*falso positivo*) representa apenas desperdício recuperável de banda. Essa assimetria de custo (Elkan, 2001) impõe uma política de decisão conservadora: na dúvida, transmitir.

As restrições operacionais que tornam o problema desafiador são:

Memória O ESP32-S3-WROOM-1-N16R8 dispõe de 512 KB de SRAM interna e 8 MB de PSRAM externa. A arena de ativações do TensorFlow Lite for Microcontrollers (TFLM) cresce com a resolução de entrada e profundidade das camadas convolucionais, podendo exceder a SRAM disponível (Lin *et al.*, 2020).

Cômputo O processador Xtensa LX7 dual-core a 240 MHz não possui NPU dedicada nem acelerador de convolução, ao contrário das plataformas Myriad 2 (Giuffrida *et al.*, 2020) e STM32N6 (Capogrosso; Magno, 2026) utilizadas em trabalhos análogos.

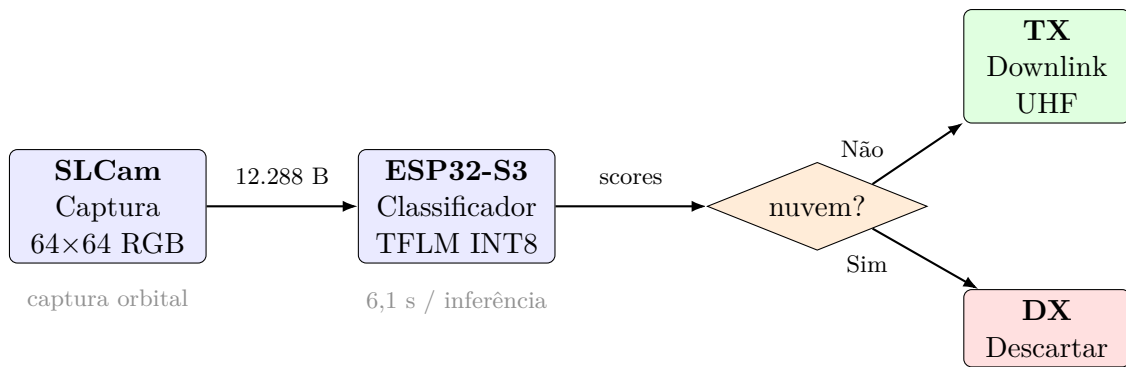
Latência O sistema opera no modelo de *decisão pós-captura*, não em tempo real: uma imagem é capturada, classificada e a decisão é tomada antes do próximo ciclo de captura. A cadência de captura orbital é tipicamente de dezenas de segundos, tornando latências da ordem de segundos aceitáveis.

Acurácia O classificador deve operar com Recall elevado para a classe “transmitir” (minimizar falsos negativos), aceitando Precision menor quando necessário para garantir a conservação de dados.

Energia Cada inferência deve ter custo energético inferior à energia poupada pelo descarte das imagens de nuvem que o sistema corretamente filtra; caso contrário, o processamento embarcado não se justifica energeticamente.

A Figura 1 sintetiza o fluxo de triagem proposto neste trabalho.

Figura 1 – Fluxo de triagem embarcada de imagens de observação da Terra



Fonte: Elaborado pelo autor.

1.3 OBJETIVOS

1.3.1 Objetivo Geral

Desenvolver e validar um sistema de classificação embarcada de imagens de observação da Terra executável no microcontrolador ESP32-S3, capaz de apoiar a decisão automática de transmitir ou descartar imagens antes do downlink em missões de nanosatélites.

1.3.2 Objetivos Específicos

1. Curar um benchmark reproduzível de imagens de observação da Terra com quatro classes de cena (*terra*, *oceano*, *nuvem* e *mista*) a partir de datasets públicos EuroSAT (Helber *et al.*, 2019) e CloudSEN12 (Aybar *et al.*, 2022), com distribuição balanceada e manifesto de dataset documentável (Gebru *et al.*, 2021);
2. Treinar e avaliar arquiteturas CNN leves adequadas ao domínio de observação da Terra, selecionando o modelo com melhor equilíbrio entre acurácia balanceada, tamanho de modelo e embarcabilidade no ESP32-S3;

3. Exportar e validar o modelo selecionado em formato TensorFlow Lite INT8, quantificando a degradação de acurácia pós-quantização por calibração representativa (*Post-Training Quantization*, PTQ) (Jacob *et al.*, 2018);
4. Implementar a inferência via TensorFlow Lite for Microcontrollers (TFLM) no ESP32-S3 utilizando ESP-IDF, medindo a arena real de memória, a latência de inferência e o comportamento determinístico do sistema;
5. Definir e avaliar política de decisão de missão fundamentada em teoria de decisão custo-assimétrica (Elkan, 2001), balanceando Recall da classe “transmitir” e economia de downlink;
6. Estimar o impacto energético do sistema, comparando o custo de inferência com a energia poupada em transmissão, e determinar o ponto de equilíbrio (*break-even*) para as taxas de downlink típicas de CubeSats universitários.

1.4 CONTRIBUIÇÕES

As principais contribuições técnicas e científicas deste trabalho são:

1. **Benchmark curado e documentado (benchmark-v1)**: 3.264 imagens de 64×64 pixels RGB em quatro classes de cena, derivadas de EuroSAT (Helber *et al.*, 2019) e CloudSEN12 (Aybar *et al.*, 2022), com curadoria automática, splits fixos reproduzíveis (seed=42, 70%/15%/15% treino/val/teste) e manifesto rastreável segundo Gebru *et al.* (Gebru *et al.*, 2021). O conjunto fixo de comparação contém 492 amostras balanceadas (123 por classe).
2. **Classificador CNN embarcado validado em hardware real**: modelo `cnn_v2_64` quantizado INT8 (158,7 KB em Flash) executado no ESP32-S3 com acurácia balanceada de **82,1%** em 492 amostras, zero degradação pós-quantização e latência determinística de **6,1 s** (jitter de 1 ms em 492 inferências).
3. **Política de decisão S3 *Dual Threshold* formalizada**: política custo-assimétrica baseada em dois limiares simultâneos sobre confiança da predição e pontuação da classe “nuvem”, com Recall de **90,5%** para a classe “transmitir”, descartando 21,7% das imagens do conjunto de teste (107 de 492) com perda de apenas 9,5% dos casos transmissíveis.
4. **Análise energética estimada com incerteza declarada**: comparação quantitativa entre custo de inferência estimado ($\approx 8 \text{ J} \pm 30\%$) e energia de transmissão UHF (47–93 J por imagem a 4.800–9.600 bps (Mehrjardi *et al.*, 2017)), com cálculo de break-even: o sistema é energeticamente vantajoso acima de uma taxa de descarte de 17,3% a 9.600 bps e de 8,6% a 4.800 bps.

5. **Protocolo de validação desacoplado do hardware de imageamento:** interface UART bidirecional entre o ESP32-S3 e um host Python, permitindo validação completa do pipeline, incluindo as 492 inferências do conjunto de teste, sem dependência do hardware de câmera SLCam, que permanece como ponto de integração futura.

O posicionamento do trabalho na literatura é específico: entre os trabalhos analisados no levantamento bibliográfico (Tabela 4, Capítulo 4), não foi identificada a combinação simultânea de **classificação de cena em quatro classes de observação da Terra, execução em MCU comercial *off-the-shelf* sem NPU, política de decisão transmitir/descartar explicitamente avaliada e benchmark de hardware real com 492 amostras.**

1.5 DELIMITAÇÃO DO ESCOPO

Para garantir a defendibilidade e a concretude do trabalho, o escopo foi deliberadamente delimitado:

Dentro do escopo Classificação de cena em nível grosseiro; treinamento, avaliação e quantização de CNNs leves; execução de inferência no ESP32-S3; medição de latência e arena de memória; política de decisão de missão; estimativa energética com incerteza declarada.

Fora do escopo Detecção de objetos; segmentação semântica como objetivo central; integração física com o hardware SLCam (tratado como extensão futura); medição direta de consumo de corrente no ESP32-S3 durante inferência (requer instrumentação INA219 ou equivalente, identificada como trabalho futuro); análise orbital completa, planejamento de missão e telemetria além do necessário para a prova de conceito; modelos treinados em GPU/TPU de grande escala.

A integração com o SLCam é uma extensão natural dos resultados, mas não é pré-requisito para validar a contribuição principal: demonstrar que um MCU comercial de baixo custo pode classificar imagens de observação da Terra com acurácia e latência operacionalmente relevantes, executar política de decisão conservadora, e estimar sua viabilidade energética.

Cabe destacar que o presente trabalho implementa e valida o **componente de classificação embarcada**, e não um sistema completo e autônomo de decisão de downlink. A decisão final de transmitir ou descartar uma imagem em uma missão operacional real depende de fatores adicionais que estão fora do escopo deste trabalho: conhecimento da posição orbital do satélite, objetivos específicos da missão (p.ex., região geográfica de interesse), estado do sistema de controle de atitude (ADCS), orçamento de energia

disponível na passagem e política de aquisição de imagens adotada. O componente de classificação desenvolvido constitui um elemento habilitador dentro de uma arquitetura de missão mais ampla, cuja especificação e integração permanecem como trabalho futuro.

1.6 ORGANIZAÇÃO DO TRABALHO

O restante deste trabalho está organizado da seguinte forma:

Capítulo 2 — Fundamentação Teórica Apresenta os fundamentos necessários para compreender e contextualizar o trabalho: o problema do downlink em CubeSats e a estratégia de triagem embarcada; TinyML em microcontroladores, incluindo restrições de memória e energia; quantização de redes neurais (PTQ INT8 (Jacob *et al.*, 2018)); classificação de cena em sensoriamento remoto e os datasets utilizados (Helber *et al.*, 2019; Aybar *et al.*, 2022); trabalhos análogos em nanosatélites; teoria de decisão custo-assimétrica (Elkan, 2001); métricas de avaliação para sistemas embarcados; e análise energética comparativa.

Capítulo 3 — Metodologia Descreve o pipeline experimental completo: composição e curadoria do benchmark; protocolo de treinamento, validação e avaliação; pipeline de exportação TFLite INT8; firmware de inferência no ESP32-S3; protocolo de validação em hardware via UART; e análise de estratégias de decisão de missão.

Capítulo 4 — Resultados e Discussão Apresenta e discute os resultados: avaliação offline dos modelos candidatos; caracterização de hardware (arena real, latência, determinismo); análise das estratégias de decisão e o impacto da quantização INT8 sobre a política; estimativa energética e análise de break-even; e comparação com trabalhos análogos da literatura.

Capítulo 5 — Conclusão Sintetiza as conclusões, discute as limitações do trabalho, e aponta as direções de trabalho futuro, incluindo medição direta de energia, integração com o SLCam e melhoria da acurácia da classe “mista”.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os fundamentos teóricos e conceituais necessários para compreender e situar as contribuições deste trabalho. As seções estão organizadas em ordem de dependência: primeiro o problema operacional que motiva o processamento embarcado, depois as tecnologias que o viabilizam, em seguida a tarefa de aprendizado e os dados utilizados, e por fim a teoria de decisão e as métricas de avaliação adotadas.

2.1 TRIAGEM EMBARCADA EM MISSÕES DE IMAGEAMENTO

O problema do retorno de dados em missões de imageamento de satélites de pequeno porte foi quantificado na Introdução: a disparidade entre a capacidade de captura e a de transmissão implica que a grande maioria das imagens geradas não pode ser transmitida ao solo dentro de uma passagem orbital. O processamento embarcado, entendido como a seleção ou filtragem das imagens a bordo antes do downlink, é a resposta estrutural a essa limitação.

2.1.1 Do Processamento por Regras ao Aprendizado de Máquina

As primeiras abordagens de filtragem embarcada eram baseadas em regras heurísticas: limites de histograma de pixel, detecção de bordas simples ou limiares de luminância para identificar imagens saturadas ou completamente escuras. Essas abordagens são computacionalmente eficientes, mas carecem de robustez para cenários com variações de iluminação, ângulo solar e composição de cena complexa (Tran *et al.*, 2004).

O experimento ASE (*Autonomous Sciencecraft Experiment*), conduzido por Tran *et al.* a bordo do satélite EO-1 da NASA em 2004, representou a primeira demonstração operacional de seleção autônoma de dados científicos em órbita (Tran *et al.*, 2004). O sistema combinava planejamento simbólico com detecção de eventos geofísicos (vulcões, inundações, incêndios) usando heurísticas e limiares programados a bordo, reduzindo o volume de downlink em mais de 70% sem perda de dado científico relevante. O ASE estabeleceu o paradigma de triagem embarcada que motiva o presente trabalho, mas operava com regras fixas sem capacidade de aprender com os dados.

A transição para aprendizado de máquina embarcado consolidou-se com o CloudScout (Giuffrida *et al.*, 2020), embarcado no satélite PhiSat-1 da ESA em 2020. O sistema executa uma rede neural convolucional em um acelerador Intel Myriad 2 (VPU com 1 TOPS de capacidade) para classificar imagens hiperespectrais em “nublada” ou “não nublada” antes da transmissão. A principal vantagem em relação às abordagens por regras

é a robustez: a CNN aprende características espectrais e texturais de nuvens diretamente dos dados, sem necessidade de codificação manual de heurísticas para cada variação de cobertura.

A progressão temporal dos sistemas de processamento embarcado em satélites de observação da Terra é apresentada na Tabela 1.

Tabela 1 – Evolução histórica do processamento embarcado em missões de imageamento

Missão	Ano	Abordagem	Tarefa	Redução de downlink
EO-1/ASE (Tran <i>et al.</i> , 2004)	2004	Regras + planejamento	Priorização geral	> 70%
PhiSat-1/CloudScout (Giuffrida <i>et al.</i> , 2020)	2020	CNN (Myriad 2)	Filtro binário nuvem	Proporcional à cobertura
BIRDS-3/CubeSatNet (Maskey; Cho, 2020)	2020	CNN (MCU ARM)	Classificação binária	~ 67% (estimado)
Phi-sat-2/WorldFloods (Mateo-Garcia <i>et al.</i> , 2023)	2023	U-Net (Myriad X)	Segmentação inundação	Apenas mapas transmitidos
VERTECS (Chatar <i>et al.</i> , 2024)	2024	CNN (RPi CM4)	Prioridade estelar	N/D
Este TCC	2026	CNN INT8 (ESP32-S3)	4 classes EO	21,7% (S3 DT)

Fonte: Elaborado pelo autor com base nas referências citadas.

2.1.2 Classificação de Cena como Tarefa de Triagem

A escolha da **classificação de cena em nível grosseiro** como tarefa central deste trabalho é motivada por três critérios: (1) a decisão de transmitir ou descartar uma imagem requer apenas uma classificação semântica de alto nível, não localização precisa de objetos; (2) a classificação de cena tem custo computacional significativamente menor que detecção de objetos ou segmentação semântica, sendo mais adequada a MCUs sem NPU; (3) para o domínio de interesse (imagens contendo terra, oceano, nuvem ou cena mista), as características discriminativas operam em escalas espaciais grandes, compatíveis com a resolução de entrada de 64×64 pixels (Lima; Marfurt, 2020).

2.2 TINYML EM MICROCONTROLADORES EMBARCADOS

TinyML é o campo que investiga a execução eficiente de modelos de aprendizado de máquina em dispositivos com restrições severas de recursos: memória da ordem de dezenas a centenas de kilobytes, potência da ordem de miliwatts a centenas de miliwatts, e capacidade de cômputo da ordem de centenas de megahertz (Lin *et al.*, 2020). Dispositivos típicos desta categoria incluem MCUs ARM Cortex-M, RISC-V e processadores Xtensa como o ESP32-S3.

2.2.1 Restrições de Memória: Pesos vs. Arena de Ativações

O insight central da literatura TinyML é que o fator limitante de memória em MCUs não são os parâmetros do modelo (pesos), mas a **arena de ativações**: a região de RAM necessária para alocar os tensores intermediários durante a inferência (Lin *et al.*, 2020).

Os pesos de um modelo quantizado INT8 são armazenados diretamente na Flash e acessados via *memory mapping* durante a inferência, sem necessidade de cópia para RAM. A arena, ao contrário, precisa residir inteiramente na RAM disponível, pois seus valores mudam a cada camada durante o passo de inferência. Em uma CNN com camadas convolucionais profundas, a arena é dominada pelo maior mapa de ativações intermediário: para o modelo `cnn_v2_64` deste TCC, a primeira camada Conv2D gera um mapa de $64 \times 64 \times 64 = 262.144$ valores de um byte cada ($= 256$ KB), dominando o orçamento de arena.

Lin et al. (Lin *et al.*, 2020) propuseram o MCUNet, um framework de co-design entre arquitetura de rede neural e sistema operacional para MCUs com 256 KB de SRAM. Através de busca neural de arquitetura (NAS) orientada explicitamente pelas restrições de arena, os autores demonstraram acurácia de 70,7% no ImageNet em dispositivos com menos de 1 MB de Flash. O resultado central é que a minimização da arena, e não da contagem de parâmetros, é a variável de projeto crítica para MCUs de baixo custo.

2.2.2 Benchmarking e Protocolos de Avaliação

O **MLPerf Tiny Benchmark** (Banbury *et al.*, 2021) estabelece um protocolo padronizado para avaliação de sistemas TinyML, definindo quatro tarefas de referência: detecção de pessoa (Visual Wake Words), classificação de imagem de baixa resolução, detecção de palavras-chave e detecção de anomalias. As métricas primárias do benchmark são: acurácia de classificação, latência de inferência de ponta a ponta, consumo de energia por inferência e pegada de memória (Flash + RAM em operação).

A relevância do MLPerf Tiny para este TCC é dupla: (1) o protocolo de medição de latência (inferência única, sem batching, com entrada real) é adotado diretamente; (2) a hierarquia de métricas, que trata acurácia, latência e energia como igualmente primárias, fundamenta a seleção de modelo apresentada no Capítulo 4.

2.2.3 Estratégias de Otimização para MCUs

Brockmann e Schlippe (Brockmann; Schlippe, 2024) investigam sistematicamente estratégias de otimização de CNNs para MCUs ARM Cortex-M, comparando quantização inteira (INT8), *pruning* estruturado e destilação de conhecimento. Os autores identificam que a combinação de **quantização INT8 com convoluções depthwise-separáveis** oferece o melhor equilíbrio entre compressão e acurácia para classificação de imagens em MCUs, com redução de memória de $4-8\times$ e aceleração de latência de $2-4\times$ em relação ao float32.

O trabalho de Capogrosso e Magno (Capogrosso; Magno, 2026), aceito para ICCPS 2026 e o mais diretamente comparável ao presente TCC, demonstra a aplicação de *pruning* estruturado iterativo combinado com quantização INT8 e mapeamento *hardware-aware*

no STM32N6 (ARM Cortex-M55 com NPU Ethos-U55) para classificação de imagens de observação da Terra em CubeSats. Os resultados reportados, 0,68–6,45 mJ por inferência e latência de 3,22–30,38 ms, estabelecem o estado da arte para esta tarefa em MCUs com acelerador dedicado.

2.2.4 O ESP32-S3 no Ecossistema TinyML

O ESP32-S3 é um SoC (*System-on-Chip*) da Espressif Systems com processador dual-core Xtensa LX7 a 240 MHz, 512 KB de SRAM interna, suporte a PSRAM externa de até 8 MB e 16 MB de Flash (Espressif Systems, 2024). Diferentemente do STM32N6 (Capogrosso; Magno, 2026) e do Intel Myriad 2 (Giuffrida *et al.*, 2020), o ESP32-S3 não possui unidade de aceleração neural (NPU). Toda a inferência é executada na CPU de propósito geral, com suporte a instruções SIMD (*Single Instruction, Multiple Data*) limitadas.

A revisão de Lasseur *et al.* (Lasseur *et al.*, 2024) posiciona MCUs de propósito geral como o ESP32-S3 na categoria de menor custo e menor consumo estático do espectro de hardware para inferência neural embarcada em satélites, ao custo de latência e consumo dinâmico substancialmente maiores que aceleradores dedicados. Essa troca é aceitável em missões com cadência de captura de dezenas de segundos por imagem, onde latências da ordem de segundos são operacionalmente admissíveis.

É importante ressaltar, contudo, que o ESP32-S3 é um componente comercial (*commercial-off-the-shelf*, COTS) não qualificado para o ambiente espacial: não possui certificação de tolerância a radiação, mecanismos formais de recuperação de falhas por eventos de partículas (SEU, *Single Event Upsets*) nem histórico de operação em vácuo com ciclagem térmica orbital. Existem arquiteturas de processamento computacionalmente equivalentes com maior adequação ao ambiente espacial (como o STM32N6 com NPU Ethos-U55 (Capogrosso; Magno, 2026) ou equivalentes com conformidade a normas ECSS), que reduziriam também a latência e o consumo em duas a três ordens de grandeza. A adoção do ESP32-S3 neste trabalho é motivada pela validação da **metodologia** em plataforma acessível e de baixo custo, e não pela intenção de utilizar esse processador diretamente em uma missão espacial real (veja os pré-requisitos operacionais na Seção 5.3).

2.3 QUANTIZAÇÃO DE REDES NEURAIS

A quantização de redes neurais consiste em representar pesos e ativações com precisão numérica reduzida em relação ao formato padrão *float32* (32 bits por valor). O objetivo é duplo: reduzir o tamanho do modelo em Flash e acelerar a inferência em hardware sem unidade de ponto flutuante de alto desempenho.

2.3.1 Fundamentos: Quantização Afim

Jacob et al. (Jacob *et al.*, 2018) propuseram o framework canônico de **quantização afim** (*affine quantization*) para inferência inteira, formalizado pela equação:

$$r = S \cdot (q - Z) \quad (2.1)$$

onde $r \in \mathbb{R}$ é o valor real (*float32*), $q \in \mathbb{Z}$ é o valor quantizado (*int8*, no intervalo $[-128, 127]$), $S \in \mathbb{R}^+$ é o fator de escala (*float32*) e $Z \in \mathbb{Z}$ é o ponto zero (*int8*). Os parâmetros S e Z são determinados por camada, permitindo ajuste fino da faixa dinâmica de representação para cada tensor.

Esta formulação permite que toda a inferência seja executada com operações inteiras de 8 bits, sem necessidade de aritmética de ponto flutuante nas operações de convolução e multiplicação matricial. O resultado é:

1. **Redução de memória em Flash:** os pesos são armazenados como *int8* (1 byte/valor), reduzindo o tamanho do modelo em $4\times$ em relação ao *float32*.
2. **Aceleração de inferência:** operações MAC (*multiply-accumulate*) em *int8* executam mais rapidamente em CPUs sem FPU dedicada, pois evitam conversões de formato e acessos ao coprocessador de ponto flutuante.
3. **Redução de energia:** operações inteiras consomem menos energia por ciclo que operações de ponto flutuante em CPUs de propósito geral.

2.3.2 Quantização Pós-Treinamento (PTQ)

A **Post-Training Quantization** (PTQ) é a estratégia de quantização adotada neste TCC. Na PTQ, o modelo é treinado completamente em *float32* e quantizado posteriormente, sem novo ciclo de treinamento. O processo utiliza um **conjunto de calibração** (tipicamente 100–1.000 amostras representativas do domínio) para determinar os valores de S e Z por camada, ajustando a faixa dinâmica de quantização à distribuição real das ativações.

Jacob et al. (Jacob *et al.*, 2018) demonstram que a PTQ preserva a acurácia do modelo original quando a distribuição de ativações por camada é aproximadamente unimodal e simétrica — condição satisfeita por CNNs treinadas com *Batch Normalization* (BN) (Ioffe; Szegedy, 2015), pois o BN normaliza explicitamente as ativações para média zero e variância unitária antes de cada camada. O modelo `cnn_v2_64` deste TCC utiliza BN em todas as camadas convolucionais, tornando-o estruturalmente adequado à PTQ (a arquitetura completa é apresentada na Figura 4).

A alternativa, *Quantization-Aware Training* (QAT), simula a quantização durante o treinamento através de operadores de quantização diferenciáveis (*fake quantization*),

geralmente resultando em menor degradação de acurácia que PTQ, mas ao custo de um ciclo completo adicional de treinamento. Para o modelo deste TCC, a PTQ produziu degradação nula, tornando a QAT desnecessária.

2.3.3 Resultados de Quantização no Presente Trabalho

A conversão PTQ via `tf.lite.TFLiteConverter` (TensorFlow 2.16) com representante de calibração de 200 amostras do conjunto de validação produziu os seguintes resultados (detalhados no Capítulo 4):

- Tamanho do modelo *float32*: 582,4 KB (em formato TFLite)
- Tamanho do modelo INT8: **158,7 KB** (redução de $3,7\times$)
- Acurácia balanceada *float32*: 0,8211 (492 amostras do test set)
- Acurácia balanceada INT8: **0,8211**, sem degradação
- Concordância de predições *float32*/INT8: 99,6%

A degradação nula é consistente com as previsões teóricas de Jacob et al. para modelos com BN em todas as camadas (Jacob *et al.*, 2018) e indica que a quantização INT8 é adequada para este domínio experimental de classificação de cena em observação da Terra, onde a variabilidade espectral entre classes é suficientemente alta para não ser mascarada pela perda de precisão numérica.

2.4 CLASSIFICAÇÃO DE CENA EM IMAGENS DE SATÉLITE

2.4.1 Caracterização da Tarefa

A **classificação de cena em sensoriamento remoto** (RSSC, do inglês *Remote Sensing Scene Classification*) consiste em atribuir um rótulo semântico a uma imagem completa, representando o tipo de superfície ou uso do solo dominante na cena. Diferentemente da detecção de objetos (que localiza instâncias precisas) e da segmentação semântica (que classifica cada pixel), a RSSC opera no nível de patch e produz uma única etiqueta por imagem.

De Lima e Marfurt (Lima; Marfurt, 2020) apresentam uma revisão sistemática de métodos de RSSC, comparando abordagens baseadas em descritores manuais (HOG, GLCM, histograma de cores), máquinas de vetores de suporte (SVM) com kernels especializados e redes neurais convolucionais (CNNs). Os autores demonstram que CNNs superam sistematicamente as abordagens clássicas, mesmo com arquiteturas relativamente simples, graças à capacidade de aprender representações hierárquicas específicas do domínio

diretamente dos dados. A diferença de desempenho é especialmente pronunciada em classes com textura complexa (como cenas mistas terra-nuvem) e em cenários com variação de iluminação.

Para o presente trabalho, as quatro classes de cena adotadas são:

- terra** Superfície terrestre sem cobertura de nuvens: vegetação, solo exposto, culturas, floresta. Dado científico de interesse primário para monitoramento ambiental.
- oceano** Superfície marítima ou lacustre. Dado de interesse para monitoramento oceânico, deriva de geleiras e climatologia.
- nuvem** Cobertura de nuvens compacta obscurecendo completamente a superfície ($> 75\%$ de cobertura). Classe descartável: não contém dado de superfície recuperável.
- mista** Cena com cobertura parcial de nuvens (25–75%), com superfície parcialmente visível. Classificada como transmissível: pode conter dado útil mesmo com cobertura parcial.

O limiar de 25–75% para a classe **mista** é derivado da metodologia de anotação do CloudSEN12 (Aybar *et al.*, 2022) e representa um compromisso operacional: imagens com menos de 25% de nuvem têm superfície suficiente para análise; imagens com mais de 75% de nuvem raramente contêm superfície analisável. Imagens na faixa intermediária são conservativamente classificadas como transmissíveis.

2.4.2 Dataset EuroSAT

O **EuroSAT** (Helber *et al.*, 2019) é um dataset de 27.000 imagens multiespectrais Sentinel-2 (European Space Agency (ESA), 2022) classificadas em 10 categorias de uso e cobertura do solo (LULC — *Land Use Land Cover*): AnnualCrop, Forest, HerbaceousVegetation, Highway, Industrial, Pasture, PermanentCrop, Residential, River e SeaLake. Cada imagem tem resolução espacial de 64×64 pixels e cobertura espectral de 13 bandas, das quais as três bandas RGB são utilizadas neste trabalho.

A escolha do EuroSAT é metodologicamente justificada por três critérios: (1) imagens Sentinel-2 são capturadas de órbita LEO e são representativas das condições ópticas de câmeras de nanosatélites universitários; (2) a resolução nativa de 64×64 pixels coincide com a resolução de entrada do modelo, eliminando interpolação; (3) o dataset é amplamente utilizado na literatura de RSSC, com DOI verificado e metodologia de coleta documentada (Helber *et al.*, 2019).

No presente trabalho, as classes do EuroSAT são remapeadas para o esquema de quatro classes da missão conforme a Tabela 2.

As classes Highway, Industrial, Residential e River são excluídas pois representam infraestrutura urbana que não se enquadra nas categorias de missão definidas. Após

Tabela 2 – Remapeamento das classes EuroSAT para as classes da missão

Classe EuroSAT	Classe da missão	Imagens
AnnualCrop	terra	3.000
Forest	terra	3.000
HerbaceousVegetation	terra	3.000
Pasture	terra	2.000
PermanentCrop	terra	2.500
SeaLake	oceano	3.000
Highway, Industrial, Residential, River	<i>excluídas</i>	—

Fonte: Adaptado de Helber *et al.* (2019).

remapeamento e amostragem balanceada para o benchmark, contribuem com 816 amostras de **terra** e 816 amostras de **oceano**.

2.4.3 Dataset CloudSEN12

O **CloudSEN12** (Aybar *et al.*, 2022) é um dataset global de 49.400 patches Sentinel-2 anotados em nível de pixel para nuvem, sombra de nuvem e superfície clara, desenvolvido por Aybar et al. Cada patch tem resolução de 512×512 pixels com 12 bandas multiespectrais.

Para o presente trabalho, os patches do CloudSEN12 são redimensionados para 64×64 pixels e reclassificados em nível de cena pela fração de cobertura de nuvem (f_{nuvem}) nas anotações pixel-a-pixel:

$$\text{classe} = \begin{cases} \text{nuvem} & \text{se } f_{\text{nuvem}} > 0,75 \\ \text{mista} & \text{se } 0,25 \leq f_{\text{nuvem}} \leq 0,75 \\ \text{terra} & \text{se } f_{\text{nuvem}} < 0,25 \text{ e terrestre} \\ \text{oceano} & \text{se } f_{\text{nuvem}} < 0,25 \text{ e marítimo} \end{cases} \quad (2.2)$$

Após o remapeamento, o CloudSEN12 contribui com 1.100 amostras de **nuvem** e 816 amostras de **mista** para o benchmark. A categoria **oceano** proveniente do CloudSEN12 produziu apenas 2 amostras válidas, indicando o viés estrutural do dataset para cenas terrestres; a totalidade da classe **oceano** é, portanto, proveniente do EuroSAT.

2.4.4 Benchmark Curado: Composição Final

A combinação EuroSAT + CloudSEN12 segue a prática de *multi-source benchmarks* descrita por Gebru et al. (Gebru *et al.*, 2021), onde múltiplas fontes com anotações heterogêneas são harmonizadas em um esquema de rótulos unificado através de um manifesto documentável. O manifesto do presente benchmark (**benchmark-v1.csv**) registra, para cada imagem, a fonte de origem, a classe original, a classe remapeada e o split de

avaliação (treino/validação/teste). A curadoria foi automatizada e não incluiu edição manual de amostras individuais, preservando reprodutibilidade e evitando intervenção subjetiva sobre o conjunto experimental.

A composição final do benchmark é apresentada na Tabela 3.

Tabela 3 – Composição do benchmark curado (`benchmark-v1.csv`)

Classe	Fonte	Total	Treino	Validação	Teste
terra	EuroSAT	816	571	122	123
oceano	EuroSAT	816	571	122	123
nuvem	CloudSEN12	816	571	122	123
mista	CloudSEN12	816	571	122	123
Total	—	3.264	2.284	488	492

Fonte: Elaborado pelo autor.

O dataset é balanceado por construção (816 amostras por classe) com splits reprodutíveis determinados por `seed=42`. Por derivar de fontes públicas e rotulagem automática por regras de remapeamento, o benchmark deve ser interpretado como base experimental reprodutível para validação de sistema embarcado, não como dataset definitivo isento de ambiguidade visual ou de casos limítrofes entre classes.

2.5 TRABALHOS ANÁLOGOS EM NANOSATÉLITES

A Tabela 4 posiciona o presente trabalho em relação aos sistemas de classificação embarcada em nanosatélites mais diretamente comparáveis na literatura. A análise revela um padrão consistente: os trabalhos anteriores ou utilizam hardware de maior capacidade (Myriad 2, RPi CM4, FPGA), ou restringem-se a tarefas binárias, ou não reportam métricas de energia. Dentro do conjunto de trabalhos analisado, o presente TCC se diferencia por combinar quatro classes de observação da Terra, MCU comercial COTS sem NPU, política de decisão TX/DX explicitamente avaliada e benchmark de hardware real com 492 amostras.

CloudScout (Giuffrida *et al.*, 2020) é o trabalho pioneiro mais próximo na abordagem: CNN para filtragem antes do downlink em missão real. A diferença central é o hardware (Myriad 2 com ≈ 1 TOPS contra CPU Xtensa sem NPU) e a tarefa (binária contra quatro classes). CloudScout não reporta latência em milissegundos nem consumo de energia por inferência, lacunas identificadas pelos próprios autores como trabalho futuro.

CubeSatNet (Maskey; Cho, 2020) é o trabalho mais análogo em termos de hardware (MCU ARM Cortex de baixo custo) e contexto (nanosatélite universitário 1U, BIRDS-3). A tarefa binária (imagem útil/inútil) é mais simples que as quatro classes deste TCC, mas a validação com dados on-orbit reais é uma vantagem metodológica. A ausência de latência e energia reportadas é uma lacuna que o presente trabalho endereça.

Tabela 4 – Comparativo de sistemas de classificação embarcada em nanosatélites

Trabalho	Hardware	Classes	Bal. Acc.	Latência	Política TX/DX
Este TCC	ESP32-S3 + PSRAM 8 MB	4	82,1%	6.103 ms	Sim (S3 Dual Threshold)
CloudScout (Giuffrida <i>et al.</i> , 2020)	PhiSat-1 Myriad 2	2	>90%	N/D	Sim (binário)
CubeSatNet (Maskey; Cho, 2020)	ARM Cortex MCU	2	90%	N/D	Implícita
VERTECS (Chatar <i>et al.</i> , 2024)	RPi CM4 (6U)	2	≈100%	N/D	Sim (prioridade)
Alainsat-1 (Cabeza-Leiva <i>et al.</i> , 2022)	FPGA 3U	2	≈85%	N/D	Sim (pixel-level)
TinyML CubeSat (Capogrosso; Magno, 2026)	STM32N6 + NPU	var.	var.	3–30 ms	Não reportada
WorldFloods (Mateo-Garcia <i>et al.</i> , 2023)	SpaceCloud Myriad X	+ N/A	IoU var.	>1 s	Implícita (mapas)

Fonte: Elaborado pelo autor com base nas referências citadas.

Capogrosso e Magno (Capogrosso; Magno, 2026) representam o estado da arte em eficiência energética (0,68–6,45 mJ/inf) e latência (3–30 ms) para esta tarefa, ao custo de hardware com NPU dedicada (STM32N6 + Ethos-U55). Essa comparação é central para contextualizar os resultados do presente trabalho: a diferença de aproximadamente três ordens de grandeza em energia é atribuída à ausência de NPU no ESP32-S3 e ao uso de PSRAM em vez de SRAM interna, conforme discutido na Seção 2.8.

2.6 TEORIA DE DECISÃO CUSTO-ASSIMÉTRICA

A transformação de uma predição probabilística de modelo (um vetor de pontuações de classe) em uma ação operacional discreta (TRANSMITIR ou DESCARTAR) é formalmente tratada pela **teoria de decisão com custo assimétrico** (Elkan, 2001).

2.6.1 Framework de Decisão Ótima sob Custos Assimétricos

Elkan (Elkan, 2001) demonstra que o classificador de Bayes ótimo, quando os custos dos tipos de erro diferem, não maximiza a acurácia, mas sim a **utilidade esperada** (ou equivalentemente, minimiza o custo esperado):

$$a^* = \arg \min_{a \in \mathcal{A}} \sum_{y \in \mathcal{Y}} P(Y=y \mid \mathbf{x}) \cdot C(a, y) \quad (2.3)$$

onde $\mathbf{x}$ é o vetor de características da entrada, $\mathcal{A}$ é o conjunto de ações possíveis, $\mathcal{Y}$ é o conjunto de classes, $P(Y=y | \mathbf{x})$ é a probabilidade *a posteriori* da classe y dada a entrada $\mathbf{x}$, e $C(a, y)$ é o custo de tomar a ação a quando a classe verdadeira é y .

2.6.2 Matriz de Custo para a Missão Orbital

Para o problema TX/DX deste TCC, o espaço de ações é $\mathcal{A} = \{\text{TX}, \text{DX}\}$ e o espaço de classes relevantes é reduzido a $\mathcal{Y} = \{\text{útil}, \text{nuvem}\}$, onde “útil” agrega as classes terra, oceano e mista (decisão: TRANSMITIR) e “nuvem” corresponde à única classe descartável (decisão: DESCARTAR). A matriz de custo é apresentada na Tabela 5.

Tabela 5 – Matriz de custo para a decisão TX/DX em missão orbital

Ação	Classe verdadeira	Tipo	Consequência e custo
TRANSMITIR	útil	VP	Dado preservado — custo nulo
DESCARTAR	nuvem	VN	Descarte correto — custo nulo
TRANSMITIR	nuvem	FP	Banda desperdiçada — custo baixo (recuperável)
DESCARTAR	útil	FN	Perda irreversível de dado científico — custo muito alto

Fonte: Elaborado pelo autor com base em Elkan (2001).

A assimetria $C(\text{FN}) \gg C(\text{FP})$ deriva da irreversibilidade da perda: um dado descartado erroneamente é irrecuperável, pois o satélite revisita a mesma área orbital apenas depois de dias. Um falso positivo (imagem de nuvem transmitida) desperdiça banda, mas é um erro recuperável: a janela de downlink seguinte pode compensar o desperdício. Essa assimetria de custo é a justificativa formal para a política conservadora adotada.

2.6.3 Limiar Ótimo de Decisão

Para a decisão binária TX/DX com custos $C(\text{FP})$ e $C(\text{FN})$, Elkan (Elkan, 2001) demonstra que o limiar de decisão ótimo θ^* , definido como o valor mínimo de $P(\text{nuvem} | \mathbf{x})$ necessário para a ação DESCARTAR, é:

$$\theta^* = \frac{C(\text{FP})}{C(\text{FP}) + C(\text{FN})} \quad (2.4)$$

Quando $C(\text{FN})/C(\text{FP}) \rightarrow \infty$ (custo de FN muito maior que FP), o limiar tende a $\theta^* \rightarrow 0$, exigindo confiança quase total na predição de nuvem antes de descartar. Na

prática, o limiar não pode ser zero (qualquer incerteza levaria a transmitir tudo), sendo necessário escolher um valor que equilibre Recall e economia de downlink.

A política S3 *Dual Threshold* adotada neste TCC implementa operacionalmente a Equação 2.4, impondo dois limiares simultâneos antes de descartar: $P_{\text{pred}} = \text{nuvem e } \text{conf} \geq 0,40$ e $\hat{p}(\text{nuvem}) \geq 0,40$. O limiar de 0,40 foi adotado como compromisso operacional simples: impõe confiança mínima para descarte, preserva a política conservadora “quando incerto, transmite” e permite implementação direta no firmware. O Capítulo 4 analisa a sensibilidade dessa escolha e mostra que, no hardware INT8, a política é equivalente ao argmax em termos de decisões finais.

2.6.4 MScore: Métrica de Avaliação de Política

Para comparar políticas de decisão com trade-offs diferentes entre Recall e Specificity, adota-se o **MScore** (Mission Score) ponderado:

$$\text{MScore} = 0,50 \times \text{Recall} + 0,35 \times \text{Specificity} + 0,15 \times \text{Precision} \quad (2.5)$$

Os pesos refletem a hierarquia de custo da Tabela 5 e a assimetria operacional $C(\text{FN}) \gg C(\text{FP})$ formalizada por (Elkan) (Elkan, 2001): Recall recebe peso máximo (0,50) por ser a proteção direta contra FN: dado útil descartado irreversivelmente constitui a perda dominante da missão. Specificity recebe peso intermediário (0,35) por controlar o desperdício de banda de downlink, o objetivo operacional secundário. Precision recebe peso mínimo (0,15) pois, para datasets balanceados como o deste trabalho, sua informação é parcialmente redundante com a interseção de Recall e Specificity. A ponderação é consistente com práticas de engenharia de sistemas onde a perda de dado científico é o custo dominante.

2.7 MÉTRICAS DE AVALIAÇÃO PARA SISTEMAS EMBARCADOS

A avaliação completa de um sistema TinyML aplicado a uma tarefa de missão requer métricas que capturem simultaneamente a qualidade de classificação, a eficiência computacional e o impacto operacional (Banbury *et al.*, 2021).

2.7.1 Acurácia Balanceada

Em datasets com possível desequilíbrio de classes, a **acurácia balanceada** (BAcc) é preferível à acurácia simples, pois pondera igualmente o desempenho em cada classe:

$$\text{BAcc} = \frac{1}{K} \sum_{k=1}^K \frac{TP_k}{TP_k + FN_k} = \frac{1}{K} \sum_{k=1}^K \text{Recall}_k \quad (2.6)$$

onde K é o número de classes e Recall_k é a sensibilidade da classe k . Para o benchmark deste TCC, balanceado por construção (816 amostras/classe), BAcc = acurácia simples. A métrica é reportada explicitamente para facilitar comparação com trabalhos futuros em datasets potencialmente desequilibrados.

2.7.2 Intervalo de Confiança para Acurácia

Dado que o conjunto de teste tem tamanho finito ($n = 492$), a acurácia pontual é acompanhada de um **intervalo de confiança** (IC). Para a acurácia balanceada, adota-se a aproximação de Wald por classe e posterior média:

$$\text{IC}_{95\%} = \hat{p} \pm z_{0,975} \sqrt{\frac{\hat{p}(1 - \hat{p})}{n_k}} \quad (2.7)$$

onde $\hat{p}$ é a estimativa pontual do Recall da classe k , n_k é o número de amostras da classe k no conjunto de teste (123 neste TCC) e $z_{0,975} \approx 1,96$ é o quantil da distribuição normal padrão para cobertura de 95%. Para $\text{BAcc} = 0,8211$ com $n_k = 123$, o IC é aproximadamente $[0,787; 0,855]$, conforme calculado na análise estatística do Capítulo 4.

2.7.3 Latência de Inferência

Seguindo o protocolo do MLPerf Tiny (Banbury *et al.*, 2021), a latência é medida como o tempo de inferência de ponta a ponta, desde a cópia do tensor de entrada na arena até a leitura do tensor de saída, em condição de inferência única (*single-inference*) sem *pipeline* ou *batching*. O tempo é medido com resolução de microssegundos via `esp_timer_get_time()` do ESP-IDF, com overhead de medição inferior a 1 μs .

2.7.4 Arena TFLM

A **arena TFLM** é a região de memória RAM alocada exclusivamente para o interpretador TensorFlow Lite Micro, contendo: (1) tensores de ativações intermediárias, (2) *scratch buffers* dos operadores (especialmente os kernels otimizados `esp-nn`), e (3) metadados do grafo de inferência. Os pesos do modelo são armazenados na Flash e acessados via *memory mapping* durante a inferência, sem ocupar a arena.

O tamanho real da arena é obtido pelo método `interpreter->arena_used_bytes()` do TFLM após uma inferência bem-sucedida, fornecendo o valor exato utilizado pelo grafo. Este valor é o único indicador confiável do consumo de RAM em operação, pois estimativas baseadas no tamanho do arquivo TFLite sistematicamente subestimam a arena real (no

presente TCC, a estimativa inicial de ≈ 150 KB foi superada por $3,7\times$ em relação ao valor real de 551 KB).

2.7.5 Energia por Inferência

O MLPerf Tiny (Banbury *et al.*, 2021) recomenda medição direta de energia por inferência com amostragem de corrente de alta resolução (INA219 ou equivalente). Quando medição direta é inviável, como ocorre no presente TCC, estimativas baseadas em datasheet são metodologicamente aceitáveis desde que acompanhadas de declaração explícita de incerteza. A estimativa adotada ($E_{\text{inf}} \approx 8 \text{ J} \pm 30\%$) segue esta prática, conforme detalhado na Seção 2.8 a seguir.

2.8 ANÁLISE ENERGÉTICA: INFERÊNCIA VERSUS TRANSMISSÃO

A justificativa energética para processamento embarcado requer a comparação entre o custo de energia da inferência no MCU e a energia poupada pelo descarte de imagens de nuvem. Esta análise determina o **ponto de equilíbrio energético** (*break-even*) — a taxa mínima de descarte necessária para que o sistema seja energeticamente vantajoso.

2.8.1 Energia de Transmissão por Imagem

Com base nos dados de potência do sistema BIRDS-3 reportados por Mehrjardi *et al.* (Mehrjardi *et al.*, 2017), a energia por bit transmitido via rádio UHF a 4.800 bps é:

$$E_{\text{bit}} = \frac{P_{\text{TX}}}{R_{\text{TX}}} = \frac{4,56 \text{ W}}{4.800 \text{ bps}} = 9,5 \times 10^{-4} \text{ J/bit} \quad (2.8)$$

onde $P_{\text{TX}} = 4,56 \text{ W}$ é a potência total do sistema durante a transmissão (OBC + rádio + antena) e $R_{\text{TX}} = 4.800 \text{ bps}$ é a taxa de downlink UHF. Para uma imagem de 64×64 pixels RGB (98.304 bits, cf. Eq. 1.1):

$$E_{\text{TX}}(4.800) = 98.304 \times 9,5 \times 10^{-4} \approx 93,4 \text{ J} \quad (2.9)$$

A Tabela 6 apresenta E_{TX} para as principais taxas de downlink encontradas na literatura de CubeSats (Saeed *et al.*, 2019).

2.8.2 Energia de Inferência (Estimada)

Com base no datasheet do ESP32-S3 (Espressif Systems, 2024) e na latência medida no hardware:

Tabela 6 – Energia de transmissão por imagem (64×64 RGB) em função da taxa de downlink

Taxa (bps)	Tempo TX (s)	E_{TX} (J)	Cenário típico
1.200	81,9	373,4	UHF mínimo / APRS
4.800	20,5	93,4	BIRDS-3, CubeSats 1U
9.600	10,2	46,7	UHF máximo comum
1.000.000	≈0,1	≈0,45	S-band mínimo (referência)

Fonte: Elaborado pelo autor com base em Mehrjardi *et al.* (2017) e Saeed *et al.* (2019).

$$E_{\text{inf}} = P_{\text{inf}} \times t_{\text{inf}} = (I \cdot V) \times t_{\text{inf}} = (0,40 \text{ A} \times 3,3 \text{ V}) \times 6,103 \text{ s} \approx 8,1 \text{ J} \quad (2.10)$$

onde a corrente de 400 mA corresponde ao valor central da faixa 350–450 mA reportada para CPU ativa a 240 MHz com PSRAM Octal 80 MHz (Espressif Systems, 2024), e a latência de 6,103 s é a mediana obtida em 492 inferências consecutivas (detalhada no Capítulo 4). A incerteza desta estimativa é de aproximadamente $\pm 30\%$, derivada da faixa de corrente do datasheet.

A discrepância de aproximadamente três ordens de grandeza em relação ao STM32N6 (Capogrosso; Magno, 2026) (0,68–6,45 mJ/inf) é explicada por três fatores: (1) presença de NPU Ethos-U55 no STM32N6, com eficiência energética por operação MAC de 10–100× superior ao Xtensa LX7 do ESP32-S3; (2) uso de PSRAM no ESP32-S3 (latência de acesso $\sim 10\text{--}20\times$ maior que SRAM interna), impondo penalidade de latência que amplifica o consumo; (3) adição de *pruning* estruturado em (Capogrosso; Magno, 2026), reduzindo operações efetivas em 80–90%.

2.8.3 Ponto de Equilíbrio Energético

O sistema de inferência embarcada é energeticamente vantajoso quando a energia poupada pelo descarte de imagens de nuvem supera o custo total de inferência. Para N imagens com taxa de descarte R :

$$N \cdot E_{\text{inf}} < N \cdot R \cdot E_{TX} \implies R^* = \frac{E_{\text{inf}}}{E_{TX}} \quad (2.11)$$

A Tabela 7 apresenta o break-even para os cenários UHF relevantes, confrontado com duas leituras da política S3 Dual Threshold: o descarte total observado, isto é, toda imagem que deixaria de ser transmitida ($107/492 = 21,7\%$), e o descarte conservador de nuvens corretamente filtradas ($72/492 = 14,6\%$). A primeira leitura representa o efeito estritamente energético sobre o enlace; a segunda separa esse efeito do custo científico de descartar dados úteis por erro de classificação.

Tabela 7 – Análise de ponto de equilíbrio energético por taxa de downlink

Taxa	E_{TX} (J)	R^* (%)	$R_{S3,total}$ (%)	Saldo total	Saldo nuvens corretas
4.800 bps	93,4	8,6	21,7	+6,01 kJ	+2,74 kJ
9.600 bps	46,7	17,3	21,7	+1,01 kJ	-0,62 kJ

Fonte: Elaborado pelo autor com base em Mehrjardi *et al.* (2017) e dados de hardware.

2.8.4 Interpretação dos Resultados Energéticos

A análise revela que o valor primário do classificador embarcado depende da taxa de downlink da missão:

- **UHF 4.800 bps** (CubeSats 1U, BIRDS-3): tanto o descarte total (21,7%) quanto o descarte conservador de nuvens corretamente filtradas (14,6%) superam o break-even de 8,6%. Nesse cenário, a economia de transmissão compensa o custo de inferência mesmo desconsiderando os descartes incorretos.
- **UHF 9.600 bps**: o descarte total (21,7%) supera o break-even de 17,3%, mas o descarte conservador de nuvens corretamente filtradas (14,6%) fica abaixo desse limiar. Assim, o ganho estritamente energético depende de considerar todo descarte como transmissão evitada; sob a leitura conservadora, o valor primário é **redução do orçamento de downlink e priorização de dados recebidos**, não economia líquida de energia.

Esta análise é metodologicamente honesta: a limitação é declarada explicitamente e a estimativa de energia é qualificada com $\pm 30\%$ de incerteza. A medição direta de consumo de corrente durante inferência no ESP32-S3, que requer instrumento INA219 ou shunt resistivo com osciloscópio, é identificada como direção de trabalho futuro.

3 METODOLOGIA

Este capítulo descreve o *pipeline* experimental adotado para construir, avaliar e validar em hardware o sistema de classificação embarcada. A metodologia é estruturada em quatro etapas sequenciais: curadoria do dataset, treinamento e seleção de modelos, exportação e quantização, e validação em hardware real. Cada etapa é versionada e rastreável, com artefatos intermediários registrados em manifestos auditáveis.

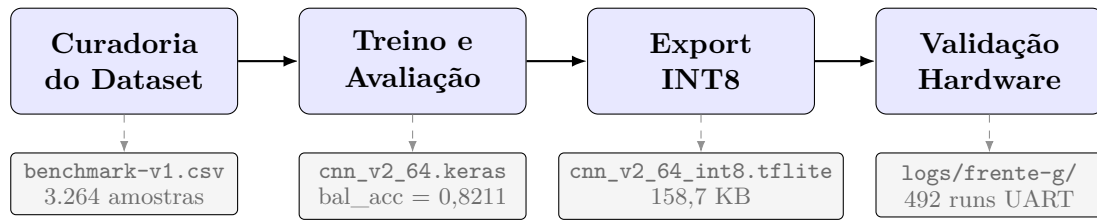
3.1 VISÃO GERAL DO PIPELINE

O *pipeline* experimental é composto por quatro estágios, executados sequencialmente e com fronteiras explícitas entre si:

1. **Curadoria automática do dataset:** coleta das fontes públicas, pré-processamento, remapeamento de classes e geração do manifesto versionado `benchmark-v1.csv`, sem modificação manual de amostras individuais, com splits de treino, validação e teste reproduzíveis (semente 42).
2. **Treinamento e avaliação offline:** sete arquiteturas avaliadas sob protocolo unificado de hiperparâmetros. A escolha do candidato embarcado combina desempenho observado, estabilidade de quantização e viabilidade de embarcamento (tamanho INT8 + arena de ativações dentro do orçamento de memória do ESP32-S3).
3. **Exportação e quantização TFLite INT8:** conversão do modelo selecionado para TensorFlow Lite com quantização pós-treinamento (PTQ) e validação da degradação de acurácia no conjunto de teste de 492 amostras.
4. **Validação em hardware real:** execução do modelo INT8 no ESP32-S3-WROOM-1-N16R8 via TensorFlow Lite Micro (TFLM) (Google LLC, 2023), com 492 amostras transmitidas via UART para medição de acurácia, latência e arena real.

A rastreabilidade é garantida pelo manifesto `benchmark-v1.csv` (documentado no Apêndice D) (Gebru *et al.*, 2021), que registra `sample_id`, fonte, classe remapeada e split para cada imagem; pelos arquivos de configuração de experimento em `experiments/history/`; e pelo *log* serial completo dos 492 runs de hardware em `logs/frente-g/`. A Figura 2 ilustra os quatro estágios e seus artefatos de saída.

Figura 2 – Pipeline experimental em quatro estágios com artefatos de rastreabilidade



Fonte: Elaborado pelo autor.

3.2 DATASET COMPOSTO

3.2.1 Estratégia Multi-Fonte

O benchmark é construído a partir de duas fontes públicas complementares, conforme a metodologia de *multi-source benchmarks* de Gebru et al. (Gebru et al., 2021):

- **EuroSAT** (Helber et al., 2019): 27.000 imagens Sentinel-2, 10 categorias LULC, resolução nativa 64×64 pixels. Fonte primária para as classes **terra** e **oceano**, cuja diversidade espectral (cinco tipos de cobertura terrestre + lago/mar) reduz o viés de domínio.
- **CloudSEN12** (Aybar et al., 2022): 49.400 *patches* Sentinel-2 anotados pixel a pixel para nuvem, sombra e superfície, cobrindo todas as regiões climáticas. Fonte primária para **nuvem** e **mista**, com anotações de qualidade verificadas pelos autores.

A escolha de fontes com anotações heterogêneas (categorias LULC no EuroSAT e fração de cobertura de nuvem no CloudSEN12) exige harmonização explícita por remapeamento de classes, descrita na Seção 3.2.2.

3.2.2 Pré-processamento e Remapeamento

O pré-processamento segue quatro camadas distintas, separando dados brutos de derivados:

1. **Raw**: dados baixados das fontes originais, sem modificação.
2. **Staged**: conversão mínima para leitura (RGB 8 bits, remoção de bordas), organização por classe de origem.
3. **Curated**: recortes quadrados, remapeamento para as quatro classes da missão, filtragem por critérios de qualidade.
4. **Benchmark**: subconjunto congelado e versionado com splits fixos.

3.2.2.0.1 Remapeamento das classes.

As classes do EuroSAT são remapeadas para o esquema da missão conforme a Tabela 2 do Capítulo 2. As classes Highway, Industrial, Residential e River são excluídas por representarem infraestrutura urbana incompatível com as categorias de missão.

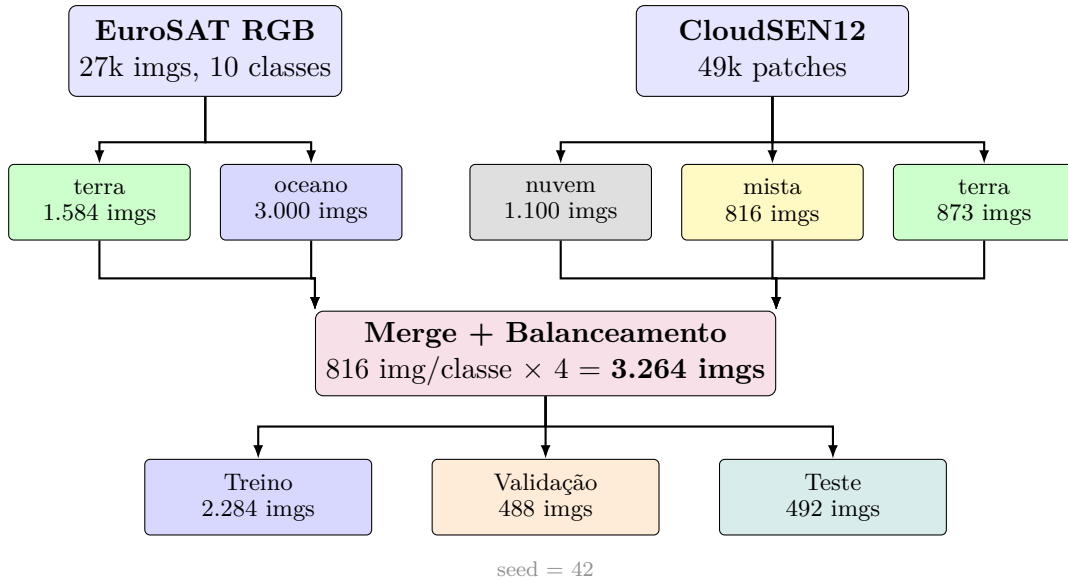
Para o CloudSEN12, a reclassificação opera sobre a fração de cobertura de nuvem f_{nuvem} nas anotações pixel a pixel (Eq. 2.2): patches com $f_{\text{nuvem}} > 0,75$ são classificados como **nuvem**; aqueles com $0,25 \leq f_{\text{nuvem}} \leq 0,75$ e fundo identificável são classificados como **mista**. Os limiares de 0,25 e 0,75 são decisão metodológica documentada, consistente com os metadados de qualidade do dataset (Aybar *et al.*, 2022).

3.2.2.0.2 Padronização de resolução.

Todas as imagens são produzidas em três resoluções derivadas: 64×64 , 96×96 e 128×128 pixels, para permitir ablação de resolução no treinamento. A resolução 64×64 é adotada como resolução principal, por coincidir com a resolução nativa do EuroSAT e minimizar a necessidade de interpolação.

A Figura 3 sintetiza o fluxo de composição do benchmark a partir das duas fontes, incluindo o remapeamento de classes e os splits fixos.

Figura 3 – Pipeline de composição do benchmark curado a partir de duas fontes públicas



Fonte: Elaborado pelo autor.

3.2.3 Composição do Benchmark e Divisão dos Conjuntos

O benchmark final contém **3.264 amostras** distribuídas em quatro classes com 816 amostras cada (Tabela 8).

Tabela 8 – Composição do benchmark `benchmark-v1.csv`

Classe	Fonte	Treino	Validação	Teste	Total
terra	EuroSAT	571	122	123	816
oceano	EuroSAT	571	122	123	816
nuvem	CloudSEN12	571	122	123	816
mista	CloudSEN12	571	122	123	816
Total	—	2.284	488	492	3.264

Fonte: Elaborado pelo autor.

O particionamento de 70 / 15 / 15% (treino/validação/teste) é gerado com `seed=42`, por *stratified split* que preserva a proporção de classes. O conjunto de **teste com 492 amostras** é mantido fixo para permitir comparação reprodutível entre modelos, quantização e execução no hardware. A interpretação dos resultados considera que o benchmark foi gerado por curadoria automática: as amostras não foram editadas manualmente, e os rótulos foram herdados das fontes originais ou dos limiares documentados de cobertura de nuvem. Assim, eventuais duplicatas visuais, amostras homogêneas ou casos limítrofes de rótulo são tratados como limitação de curadoria automática, não como intervenção manual sobre o conjunto experimental.

Essa escolha preserva a rastreabilidade e a reprodutibilidade do benchmark, mas limita a força das conclusões estritamente estatísticas sobre generalização. Por esse motivo, a contribuição principal do trabalho é interpretada em conjunto com a validação embarcada end-to-end no ESP32-S3, e não como proposta de um novo benchmark definitivo de observação da Terra.

3.3 ARQUITETURAS AVALIADAS

Sete arquiteturas são avaliadas sob o mesmo protocolo, divididas em três categorias: CNNs customizadas, modelos de *transfer learning* e uma *baseline* clássica.

3.3.1 CNN Custom (`cnn_v2`)

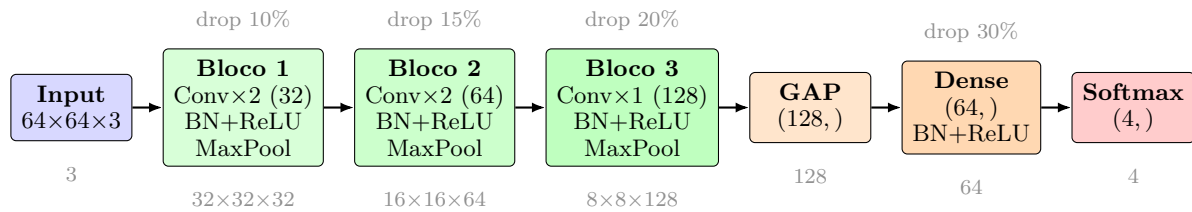
A arquitetura principal, denominada `cnn_v2`, é uma CNN customizada com **149.092 parâmetros** composta por blocos convolucionais com *Batch Normalization* (BN) e *Max-Pooling*, seguidos de camadas completamente conectadas e ativação *Softmax* na saída. O uso de BN em todos os blocos convolucionais é decisivo para dois aspectos: (1) estabilização do treinamento em presença de *data augmentation* agressiva, e (2) adequação estrutural à quantização INT8 (Jacob *et al.*, 2018), pois o BN normaliza as ativações reduzindo a variância entre camadas.

A mesma arquitetura é avaliada nas três resoluções de entrada, produzindo três variantes independentemente treinadas:

- **cnn_v2_64**: entrada 64×64 , 117 M operações MAC por inferência.
- **cnn_v2_96**: entrada 96×96 , 263 M MACs.
- **cnn_v2_128**: entrada 128×128 , 468 M MACs.

Os três modelos têm o mesmo número de parâmetros (149.092) e tamanho idêntico do arquivo INT8 (158,7 KB em Flash), pois a arquitetura é a mesma. A diferença de MACs é proporcional à área da imagem de entrada. A Figura 4 ilustra a arquitetura da variante **cnn_v2_64**. Adicionalmente, a variante **cnn_custom** (protocolo de treinamento v1, descrito adiante) é avaliada como comparador interno.

Figura 4 – Arquitetura do modelo **cnn_v2_64** — blocos convolucionais e dimensões



Fonte: Elaborado pelo autor. GAP = *GlobalAveragePooling2D*.

3.3.2 Modelos de Transfer Learning

MobileNetV2 e **EfficientNet Lite0** são avaliados com pesos pré-treinados no ImageNet, *fine-tuning* completo e cabeça de classificação substituída por camada densa de quatro classes. Ambos foram projetados para entradas de 224×224 pixels. A redução para 64×64 elimina múltiplas camadas de *strided convolution* dos estágios iniciais, degradando severamente a capacidade de extração de características. Os modelos INT8 resultantes têm tamanho superior a 5 MB (MobileNetV2: ≈ 5.353 KB; EfficientNet Lite0: ≈ 6.388 KB), tornando-os inviáveis para embarcamento no ESP32-S3 independentemente da acurácia.

3.3.3 Baseline Clássica

O **baseline_histogram_lr** é um classificador composto por histograma RGB de 256 *bins* por canal (vetor de 768 características) seguido de regressão logística multinomial (*sklearn*). Executa em qualquer MCU sem unidade de IA: operações de histograma e produto de matriz estão disponíveis em todo microcontrolador. Serve como limite inferior de desempenho aceitável: qualquer CNN deve superar esta *baseline* para justificar o custo de hardware adicional.

3.4 PROTOCOLO DE TREINAMENTO

3.4.1 Configuração Comum

Todos os modelos CNN são treinados com **Python 3.14 + Keras 3.14.1 + PyTorch 2.12.0** (backend PyTorch). A Tabela 9 apresenta os hiperparâmetros comuns a todas as variantes.

Tabela 9 – Hiperparâmetros de treinamento (protocolo v2)

Parâmetro	Valor
Batch size	32
Máximo de épocas	80
Learning rate inicial	1×10^{-3}
Warmup (épocas)	5 (linear de 1×10^{-5} até 10^{-3})
Early stopping	patience = 20 (monitor: <code>val_balanced_accuracy</code>)
ReduceLROnPlateau	patience = 5, fator = 0,5
Checkpoint	Melhor <code>val_balanced_accuracy</code>
Balanceamento	<code>class_weight</code> = {0 : 1, 0; 1 : 1, 0; 2 : 1, 0; 3 : 1, 0}
Semente	42

Fonte: Elaborado pelo autor.

A métrica de monitoramento é a acurácia balanceada no conjunto de validação (`val_balanced_accuracy`), para evitar que o critério de parada privilegie classes majoritárias em caso de desequilíbrio eventual.

3.4.2 Aumento de Dados

A *data augmentation* é aplicado apenas ao conjunto de treino, sem modificação do conjunto de validação ou teste:

- Espelhamento horizontal e vertical (probabilidade 0,5 cada).
- Rotação aleatória ($\pm 15^\circ$).
- Zoom aleatório ($\pm 10\%$).
- Variação de brilho e contraste ($\pm 10\%$).

As transformações simulam variações de ângulo de aquisição orbital e condições de iluminação solar, sem alterar as características espectrais que definem cada classe. Espelhamentos são particularmente relevantes pois imagens de satélite não têm orientação preferencial.

3.4.3 Curva de Treinamento e Early Stopping

O callback `ReduceLROnPlateau` (fator 0,5, *patience* 5) reduz o *learning rate* quando a validação estagna, forçando o otimizador a explorar vizinhanças mais finas do espaço de parâmetros. O `ModelCheckpoint` captura o melhor `val_balanced_accuracy` ao longo de todo o treinamento, mesmo que ocorra após múltiplas reduções de LR. Essa dinâmica é relevante para o `cnn_v2_64`, cujo treinamento definitivo se estendeu por 71 épocas (dentro do limite de 80) com múltiplos eventos de `ReduceLR`, exploração de regiões distintas do espaço de parâmetros e *checkpoint* final superior ao obtido com convergência mais rápida (Brockmann; Schlippe, 2024).

3.5 SELEÇÃO DE MODELO

A seleção de modelo combina critérios de desempenho e de engenharia embarcada:

1. **Desempenho de classificação:** acurácia balanceada (BAcc), métricas por classe e estabilidade após quantização INT8, reportadas sobre o conjunto fixo de comparação.
2. **Viabilidade de embarcamento:** tamanho INT8 em Flash + arena de ativações TFLM deve caber no orçamento de memória disponível no ESP32-S3.

A Tabela 10 apresenta os resultados de todos os modelos avaliados. Os valores de BAcc são usados como evidência comparativa, enquanto a decisão final de embarque prioriza viabilidade de memória, custo computacional e validação real no alvo.

Tabela 10 – Comparativo de modelos — acurácia e viabilidade de embarcamento

Modelo	Res.	BAcc	INT8 (KB)	Degradação	Embarcável?
cnn_v2_64	64	0,8211	158,7	0,000	Sim (PSRAM)
<code>cnn_v2_128</code>	128	0,8089	158,7	−0,002	Não (arena >512 KB)
<code>cnn_v2_96</code>	96	0,8069	158,7	−0,004	Sim (PSRAM, alternativa)
<code>cnn_custom</code>	64	0,7846	158,7	−0,002	Sim (PSRAM)
MobileNetV2	64	0,7602	≈5.353	N/A	Não (modelo ≫512 KB)
EfficientNet Lite0	64	0,7561	≈6.388	N/A	Não (modelo ≫512 KB)
Baseline (hist+LR)	64	0,7154	N/A	N/A	Sim (sem TFLM)

Fonte: Elaborado pelo autor. BAcc medida no conjunto fixo de comparação (492 amostras).

O modelo selecionado é o **cnn_v2_64** por apresentar o melhor compromisso de engenharia entre desempenho e embarcabilidade: BAcc de 0,8211 no conjunto fixo de comparação, degradação nula após quantização INT8, menor custo computacional (117 M MACs vs. 263 M de `cnn_v2_96`) e melhor F1 na classe mais difícil (*mista*: 0,6818 vs. 0,6320 do `cnn_v2_96`). O artefato embarcável é `experiments/tflite/cnn_v2_64_int8.tflite` (158,7 KB).

A superioridade do `cnn_v2_64` sobre variantes de maior resolução é contraintuitiva mas estruturalmente explicada: (1) a resolução 64×64 preserva a relação parâmetros/pixel favorável (36,4 vs 16,2 do 96×96); (2) o *downsampling* enfatiza a fração de cobertura de nuvem, que é o critério definatório da classe `mista` (a principal fonte de erros), em detrimento de detalhes texturais locais que degradam a separabilidade `mista`/nuvem a resoluções maiores; (3) o treinamento mais longo (71 épocas vs 54 do `cnn_v2_96`), decorrente de dinâmica exploratória com `ReduceLROnPlateau`, produziu um *checkpoint* superior.

3.6 EXPORTAÇÃO E QUANTIZAÇÃO TFLITE INT8

O modelo selecionado é exportado para TensorFlow Lite INT8 via **Post-Training Quantization** (PTQ) com **Python 3.12 + TensorFlow 2.16**. Um ambiente Python separado é necessário pois o treinamento usa Keras 3 (Python 3.14), incompatível com o método de exportação TFLite na versão 2.16: o `TFLiteConverter` da TF 2.16 requer modelos no formato `concrete_function`, obtido via `tf.function` com especificação explícita da assinatura de entrada, e não aceita modelos Keras 3 nativamente.

O procedimento de exportação segue os passos:

1. Carregar o checkpoint Keras (`cnn_v2_64_best.keras`) na TF 2.16 via compatibilidade de pesos (`tf.saved_model`).
2. Instanciar o `TFLiteConverter` via `from_concrete_functions()`.
3. Configurar PTQ com `optimizations = [OPTIMIZE_FOR_SIZE]` e definir o *representative dataset*: 200 amostras aleatórias do conjunto de validação, com entrada no formato `float32` $\in [0, 255]$ (a camada `Rescaling` interna ao modelo realiza a normalização $\times 1/255$ antes das convoluções).
4. Exportar o modelo INT8 com quantização completa de pesos e ativações.
5. Validar BAcc no conjunto fixo de comparação via interpretador TFLite.

Os resultados da exportação são:

- Tamanho `float32`: 582,4 KB (referência, sem quantização).
- Tamanho INT8: **158,7 KB** (redução de $3,7\times$).
- BAcc `float32` (TFLite): 0,8211.
- BAcc INT8 (TFLite): **0,8211**, sem degradação.
- Concordância de predições `float32`/INT8: **99,6%**.

A degradação nula é consistente com as previsões teóricas de Jacob et al. (Jacob et al., 2018) para modelos com BN em todas as camadas (Seção 2.3).

3.7 IMPLEMENTAÇÃO NO ESP32-S3

3.7.1 Hardware

O hardware alvo é o módulo **ESP32-S3-WROOM-1-N16R8** (Espressif Systems, 2024, 2023), montado na placa de desenvolvimento **ESP32-S3-DevKitC-1** (Espressif Systems). As especificações relevantes para a inferência são:

- Processador: dual-core Xtensa LX7 a 240 MHz, sem NPU.
- SRAM interna: 512 KB (bloco contíguo disponível ≈ 256 KB após inicialização do sistema).
- PSRAM: 8 MB Octal a 80 MHz (memória disponível para arena de ativações).
- Flash: 16 MB (armazena firmware + modelo via *memory mapping*).
- Interface UART: baudrate configurável até 4 Mbps (115.200 bps neste trabalho).

A arena TFLM do `cnn_v2_64` requer 551 KB de memória contígua (medida real, Capítulo 4), excedendo a SRAM interna disponível. O firmware aloca a arena na **PSRAM** via `heap_caps_malloc(MALLOC_CAP_SPIRAM)`. O modelo de 158,7 KB permanece na Flash, acessado via *memory mapping* sem ocupar RAM.

3.7.2 Pilha de Software

O firmware é desenvolvido com:

- **ESP-IDF v5.4 (LTS)**: *framework* oficial da Espressif para desenvolvimento em C++ no ESP32-S3. Fornece acesso de baixo nível ao PSRAM (`MALLOC_CAP_SPIRAM`), ao temporizador de alta resolução (`esp_timer_get_time()`, resolução de 1 μ s) e ao UART.
- **Componente espressif/esp-tflite-micro v1.3.1**: versão oficial do TensorFlow Lite Micro para ESP32, com kernels `esp-nn` otimizados com instruções SIMD (*Single Instruction, Multiple Data*) do Xtensa LX7 para operações de convolução inteira.

O modelo INT8 é convertido para *array* C via `xxd -i` e compilado diretamente na imagem de firmware. A inferência utiliza os operadores `Conv2D`, `MaxPool2D`, `Reshape`, `FullyConnected` e `Softmax` do `MicroMutableOpResolver`, sem dependências externas além do TFLM.

3.7.3 Protocolo de Comunicação UART

A interface entre o ESP32-S3 e o *host* (computador com Python) segue um protocolo ASCII bidirecional via UART a 115.200 baud, projetado para permitir validação automatizada com o conjunto de teste completo:

```
Host -> ESP32 : PING\r\n
ESP32 -> Host : ##PONG\r\n

Host -> ESP32 : INFER\r\n
ESP32 -> Host : ##READY\r\n
Host -> ESP32 : [12288 bytes RGB bruto, 64x64x3, uint8]
ESP32 -> Host : ##RES,<id>,<nome>,<conf>,<s0>,<s1>,<s2>,<s3>,<ms>,<dec>\r\n

Host -> ESP32 : STATUS\r\n
ESP32 -> Host : ##STATUS,<arena_kb>,<sram_free_kb>\r\n
```

Listing 3.1 – Protocolo UART de inferência (ESP32-S3 ↔ host Python)

Cada imagem do conjunto de teste é carregada pelo *script* Python (`scripts/infer_dataset.py`), convertida para *buffer* `uint8` de 12.288 bytes ($64 \times 64 \times 3$) e transmitida via UART após o sinal `##READY`. O ESP32-S3 responde com os *scores softmax* das quatro classes, a predição, a confiança, a latência em milissegundos e a decisão de missão (TX/DX) conforme a política S3. O formato completo da resposta e a descrição de cada campo encontram-se no Apêndice B.

3.8 PROTOCOLO DE AVALIAÇÃO EM HARDWARE

A avaliação em hardware visa verificar se o modelo INT8 embarcado produz resultados idênticos à avaliação *offline* (TFLite no computador), quantificar a latência de inferência real e medir a arena TFLM efetivamente utilizada.

O protocolo segue as diretrizes do MLPerf Tiny (Banbury *et al.*, 2021) para medição de sistemas embarcados:

1. **Verificação de conectividade:** sequência PING/PONG verifica que o firmware está operacional antes de iniciar o benchmark.
2. **Aquecimento:** 5 inferências com imagens sintéticas (zeros) são executadas e descartadas para estabilizar o estado de *cache* da PSRAM.
3. **Benchmark principal:** as 492 imagens do conjunto de teste são transmitidas em ordem determinística (mesma ordem do `splits-v1.csv`).
4. **Medição de latência:** cada inferência é cronometrada via `esp_timer_get_time()` (resolução de 1 μ s), medindo o intervalo entre a entrada em `Invoke()` e seu retorno.

5. **Arena real:** após a primeira inferência bem-sucedida, o firmware obtém o consumo real de RAM via `arena_used_bytes()` (TFLM), único indicador confiável dessa métrica em operação.

O *run* definitivo de avaliação hardware (20260520_151533) processou 492 imagens em 58,9 minutos (latência média de 6.103 ms por imagem), com jitter de apenas 1 ms (mín. = máx. = 6.103–6.104 ms), indicando comportamento determinístico do sistema sob FreeRTOS com *tick* de 1 kHz.

Os resultados de acurácia do hardware são comparados com a referência *offline* (TFLite INT8 no computador) para quantificar a concordância. Qualquer diferença acima de 0,5% seria atribuída a diferenças de implementação do operador *esp-nn* vs a implementação de referência do TFLite.

3.9 POLÍTICA DE DECISÃO DE MISSÃO

A etapa final do *pipeline* converte a predição probabilística do modelo em uma ação operacional discreta: TRANSMITIR ou DESCARTAR.

3.9.1 Derivação da Política

A política é derivada diretamente da teoria de decisão custo-assimétrica de Elkan (Elkan, 2001) (Seção 2.6). Com a assimetria $C(\text{FN}) \gg C(\text{FP})$ (descarte incorreto de dado útil é irreversível; transmissão desnecessária de nuvem é recuperável), o limiar ótimo θ^* de decisão tende a valores baixos, exigindo alta confiança na predição de nuvem antes de descartar.

Quatro famílias de estratégias são simuladas sobre os *scores softmax* das 492 amostras, com varredura (*sweep*) de parâmetros:

S0 — Argmax: predição direta sem modificação de limiar.

S1 — Confidence Threshold: transmite se confiança $< T$ (*incerto*); senão usa argmax.

S2 — Score P(nuvem) Threshold: descarta se $P(\text{nuvem}) > T$, ignorando argmax.

S3 — Dual Threshold: descarta somente se $\text{argmax} = \text{nuvem}$ e confiança $\geq T_c$ e $P(\text{nuvem}) \geq T_n$.

S4 — Margem: transmite se $P_1 - P_2 < M$ (predição ambígua).

A avaliação de cada estratégia utiliza o MScore (Eq. 2.5) como métrica de *ranking*. A Tabela 11 apresenta o comparativo das estratégias ótimas de cada família.

Tabela 11 – Comparativo das estratégias de decisão simuladas (492 amostras)

Estratégia	Recall	Spec.	Prec.	FN	FP	Vol%	MScore
Transmite tudo (sem modelo)	1,000	0,000	0,750	0	123	100%	0,613
S0 Argmax baseline	0,905	0,585	0,868	35	51	78,2%	0,788
S1* conf < 0,40 → transmite	0,911	0,585	0,868	33	51	78,7%	0,790
S2* $P(\text{nuvem}) > 0,30$	0,743	0,911	0,961	95	11	57,9%	0,834
S3* Dual ($T_n=T_c=0,40$)	0,911	0,585	0,868	33	51	78,7%	0,790
S4* margem < 0,05	0,919	0,553	0,860	30	55	80,1%	0,782

Fonte: Elaborado pelo autor. Simulação sobre *scores float32* do *run* definitivo (492 amostras).

A análise revela que a confusão entre *mista* e *nuvem* é estrutural: os *scores* $P(\text{nuvem})$ das duas classes se sobrepõem no intervalo $[0,30; 0,65]$ (média *mista* = 0,39; média *nuvem* = 0,51), tornando impossível eliminar completamente os falsos negativos sem perda significativa de recall. Estratégias agressivas como S2 com $T = 0,30$ maximizam o MScore, mas ao custo de 95 FN (25,7% do dado útil perdido), inaceitável para uma missão orbital onde cada imagem é irrepetível.

3.9.2 Política Adotada: S3 *Dual Threshold*

A estratégia recomendada é a **S3 Dual Threshold** ($T_n = 0,40$, $T_c = 0,40$):

```

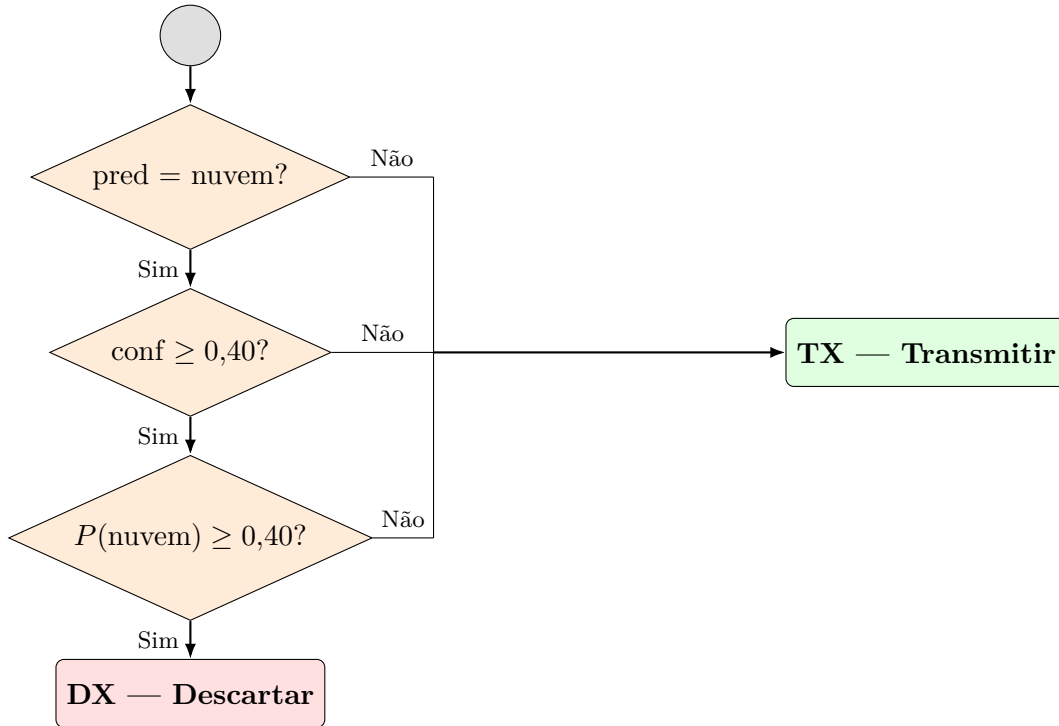
1 // S3 Dual Threshold: descartar somente com dupla evidencia de nuvem
2 static const float kDiscardThreshConf = 0.40f;
3 static const float kDiscardThreshNuvem = 0.40f;
4
5 bool mission_should_transmit(const ClassifierResult& res) {
6     // Politica: "quando incerto, TRANSMITE" (C(FN) >> C(FP))
7     // So descarta se: argmax=nuvem E confiante E P(nuvem) alta
8     if (res.class_id == 1 /* nuvem */
9         && res.confidence >= kDiscardThreshConf
10        && res.scores[1] >= kDiscardThreshNuvem) {
11         return false; // DESCARTAR
12     }
13     return true; // TRANSMITIR -- politica conservadora
14 }
```

Listing 3.2 – Implementação da política S3 no firmware do ESP32-S3

A Figura 5 ilustra o fluxo de decisão da política S3.

A política S3 implementa operacionalmente a condição $\theta^* \ll 0,5$ de Elkan (2.4): exige dupla evidência (predição argmax + dois limiares de probabilidade) antes de qualquer descarte. O custo de implementação é três comparações *float*, custo negligível frente à latência de 6.103 ms de inferência.

O limiar de 0,40 foi adotado como regra operacional simples e conservadora: exige confiança mínima antes de descartar, mas evita tornar o sistema excessivamente permissivo

Figura 5 – Fluxograma da política S3 *Dual Threshold*

Fonte: Elaborado pelo autor.

a ponto de transmitir praticamente todas as imagens. A varredura de estratégias é usada como análise de sensibilidade da política, não como otimização fina do conjunto de teste. Para dados úteis com alta confiança de nuvem (imagens *mista* com $P(\text{nuvem}) \approx 0,58$), nenhum limiar simples evita esse FN: a limitação é estrutural à separabilidade das classes no espaço espectral de 64×64 pixels RGB.

3.9.3 Validação da Política em Hardware INT8

A execução da política S3 no hardware INT8 real revelou uma discrepância em relação à simulação *float32*: as 2 imagens previstas como “salvas” pelo S3 vs S0 (FN = 33 vs 35 na simulação) continuam sendo descartadas no hardware. A causa é a discretização INT8 dos *scores softmax*: casos com confiança *float32* próxima a 0,40 têm *scores* INT8 que ainda excedem ambos os limiares, mantendo o comportamento idêntico ao S0 *argmax*.

Esta observação tem implicações práticas para a validação de sistemas: políticas de limiarização que produzem benefício marginal em *scores float32* podem ser neutralizadas pela quantização INT8. O S3 é mantido no firmware por **interpretabilidade**: implementa explicitamente a regra “quando incerto, transmite”, sem custo computacional adicional, com comportamento equivalente ao S0 *argmax* no hardware INT8.

4 RESULTADOS E DISCUSSÃO

4.1 AVALIAÇÃO OFFLINE DOS MODELOS

4.1.1 Comparativo de Acurácia

A Tabela 12 apresenta os resultados de todos os modelos avaliados no conjunto fixo de comparação (492 amostras, 4 classes balanceadas).

Tabela 12 – Comparativo de modelos — conjunto fixo de comparação ($n = 492$, $\text{seed} = 42$)

Modelo	Res.	BAcc	INT8 (KB)	Arena est.	Viável
cnn_v2_64	64	0,8211	158,7	551 KB (PSRAM)	Sim
cnn_v2_128	128	0,8089	158,7	~540 KB	Não
cnn_v2_96	96	0,8069	158,7	~310 KB	Sim*
cnn_custom	64	0,7846	158,7	551 KB (PSRAM)	Sim
MobileNetV2	64	0,7602	—	—	—
EfficientNet-Lite0	64	0,7561	—	—	—
Baseline (histograma+LR)	—	0,7154	—	—	—

Fonte: Elaborado pelo autor.

Nota: *cnn_v2_96: arena estimada por análise de camadas; não validada em hardware.

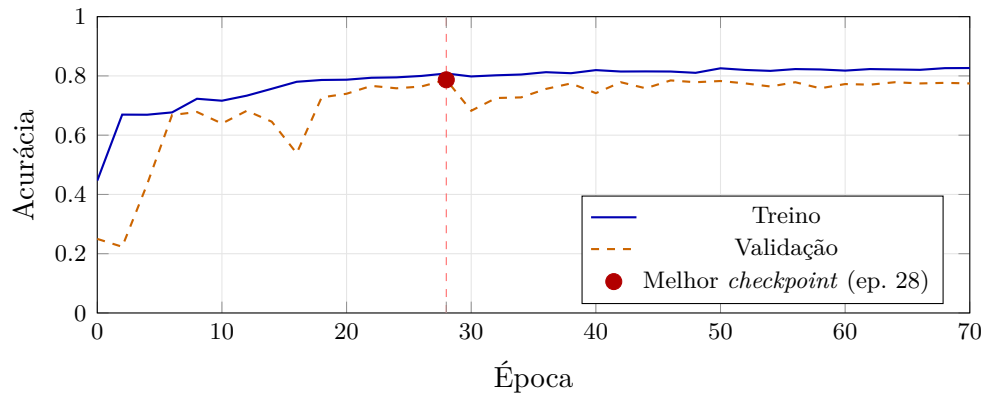
As diferenças entre as variantes *cnn_v2_** não são estatisticamente significantes ($z < 0,58$, $p > 0,56$). A seleção do *cnn_v2_64* é baseada em embarcabilidade (menor arena estimada e menor latência), não em diferença de acurácia. A superação da baseline pelo *cnn_v2_64* é estatisticamente robusta ($z = 4,15$, $p < 0,001$, $\text{IC}_{95\%}$: $[0,787; 0,855]$). Detalhes da arquitetura de camadas e pegada de memória de cada modelo encontram-se no Apêndice A; resultados completos por modelo com IC 95% no Apêndice C.

A Figura 6 apresenta a evolução da acurácia ao longo do treinamento definitivo do *cnn_v2_64* (71 épocas, Run 3). O melhor *checkpoint* ($\text{val_accuracy} = 0,787$) foi capturado na época 28 pelo *ModelCheckpoint*; o treinamento prosseguiu por mais 43 épocas com *ReduceLROnPlateau*, explorando vizinhanças mais finas do espaço de parâmetros.

4.1.2 Análise por Classe

A Tabela 13 detalha o desempenho por classe do modelo selecionado.

A confusão estrutural entre *mista* e *nuvem* é inerente à definição das classes: imagens *mista* têm $P(\text{nuvem})$ médio de 0,39 vs. 0,51 para *nuvem* pura, e as distribuições se sobrepõem em $P(\text{nuvem}) \in [0,30; 0,65]$. Nenhuma estratégia de *threshold* elimina essa confusão sem perda de dados úteis.

Figura 6 – Curva de treinamento do *cnn_v2_64* — run definitiva (71 épocas)

Fonte: Elaborado pelo autor. Dados de `experiments/history/cnn_v2_64_history.csv`.

Tabela 13 – Desempenho por classe — *cnn_v2_64*, hardware real (n = 492)

Classe	Precisão	Recall	F1	N
mista	0,636	0,740	0,684	123
nuvem	0,673	0,585	0,626	123
oceano	1,000	1,000	1,000	123
terra	0,992	0,959	0,975	123
Macro	—	0,821	0,821	492

Fonte: Elaborado pelo autor (run 20260520_151533).

4.1.3 Matriz de Confusão e Análise de Erros

A Tabela 14 apresenta a matriz de confusão do *cnn_v2_64* no hardware real (run 20260520_151533, 492 amostras INT8).

Tabela 14 – Matriz de confusão — *cnn_v2_64* em hardware INT8 (n = 492)

Real \ Previsto	mista	nuvem	oceano	terra
mista	91	32	0	0
nuvem	50	72	0	1
oceano	0	0	123	0
terra	2	3	0	118

Fonte: Elaborado pelo autor.

Nota: Diagonal em negrito: acertos. Linhas: classe real; colunas: classe predita.

A Figura 7 apresenta os mesmos dados em formato visual, evidenciando a assimetria da confusão por intensidade de cor.

A matriz de confusão revela dois padrões principais. O primeiro é a **confusão mista↔nuvem**: 32 amostras de *mista* foram preditas como *nuvem* (26,0%) e 50 amostras de *nuvem* como *mista* (40,7%). A confusão é assimétrica, com o modelo tendendo a subestimar a cobertura de nuvem, consequência direta da definição operacional gradual:

Figura 7 – Heatmap da matriz de confusão — *cnn_v2_64* em hardware INT8 ($n = 492$)

		<i>Predito</i>			
		mista	nuvem	oceano	terra
<i>Real</i>	mista	91 (74%)	32 (26%)	0	0
	nuvem	50 (41%)	72 (59%)	0	1
	oceano	0	0	123 (100%)	0
	terra	2	3	0	118 (96%)

Fonte: Elaborado pelo autor.

Nota: Verde: acertos (diagonal); vermelho: erros (intensidade proporcional ao valor). Células com valor 0 em cinza claro.

mista = 20–70% de cobertura; *nuvem* = $\geq 70\%$. Não há fronteira espectral nítida entre essas classes em imagens de 64×64 pixels RGB.

Segundo, o **alto desempenho em oceano**: 123/123 acertos ($F1 = 1,000$; $IC_{95\%}$: $[0,970; 1,000]$ via Clopper-Pearson). A assinatura espectral uniforme da água aberta (canal azul/verde dominante, baixa textura) é altamente discriminativa mesmo em baixa resolução. A classe *terra* também apresenta $F1 = 0,975$, indicando que a rede distingue com segurança os extremos do espectro cena.

Do ponto de vista da missão, os erros mais custosos são imagens de dado útil incorretamente descartadas. No total, 35 imagens foram descartadas indevidamente (FN da política S3): 32 amostras de *mista* e 3 de *terra* classificadas como *nuvem*. Nenhuma amostra de *oceano* foi descartada incorretamente. Os 51 casos opostos, imagens de *nuvem* transmitidas desnecessariamente (FP), representam custo de downlink, não de dados úteis perdidos.

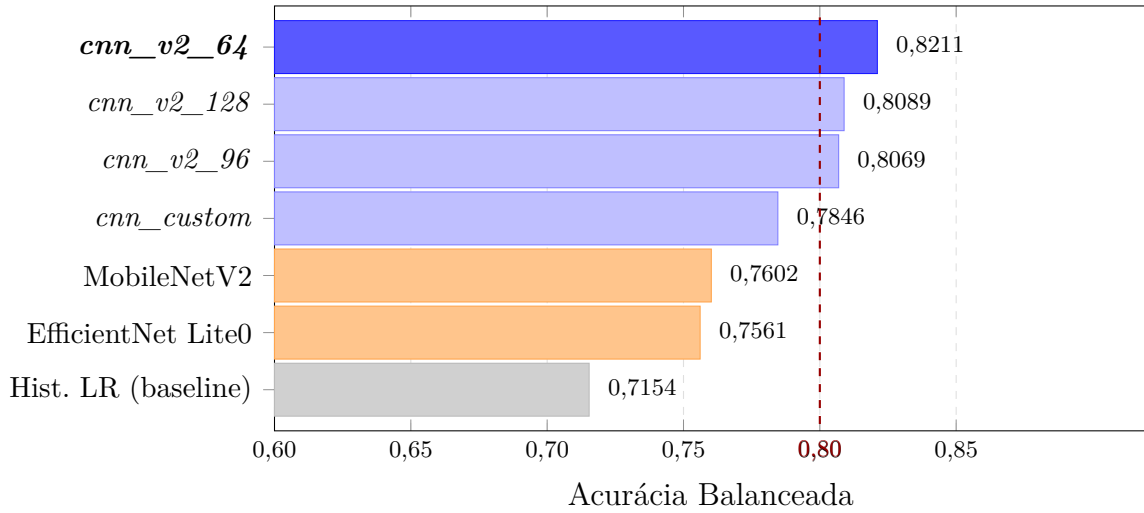
4.2 QUANTIZAÇÃO INT8 — ZERO DEGRADAÇÃO

A quantização PTQ INT8 via (Jacob *et al.*) (Jacob *et al.*, 2018) produziu:

- Redução de tamanho: 582,4 KB (float32 TFLite) $\rightarrow$ 158,7 KB (INT8), fator $3,7\times$
- Degradação de acurácia: $\Delta BAcc = 0,000$ (concordância de 99,6% entre float32 e INT8 em 492 amostras)
- Implicação: a calibração representativa foi suficiente para preservar a BAcc observada após PTQ, sem necessidade de QAT nesta prova de conceito

A Figura 8 compara a acurácia balanceada de todas as arquiteturas avaliadas, evidenciando que as CNNs treinadas do zero superam o *transfer learning* e a baseline clássica.

Figura 8 – Acurácia balanceada por arquitetura — *test set* ($n = 492$), hardware INT8



Fonte: Elaborado pelo autor.

Nota: Azul escuro: modelo adotado (*cnn_v2_64*). Linha tracejada: referência de desempenho em 0,80. Modelos ordenados do pior (baixo) para o melhor (cima).

4.3 CARACTERIZAÇÃO DE HARDWARE (ESP32-S3)

4.3.1 Arena TFLM e Memória

A arena real medida no ESP32-S3 é de **564.160 bytes (551 KB)**, excedendo a SRAM interna disponível (256 KB de bloco contíguo). A alocação em PSRAM é automática via `MALLOC_CAP_SPIRAM`.

A estimativa inicial de 150 KB estava incorreta pois confundia tamanho do modelo (pesos em Flash, 158,7 KB) com arena (ativações intermediárias em RAM). A primeira camada Conv2D sozinha produz $64 \times 64 \times 64 = 262.144$ bytes (256 KB) de ativações.

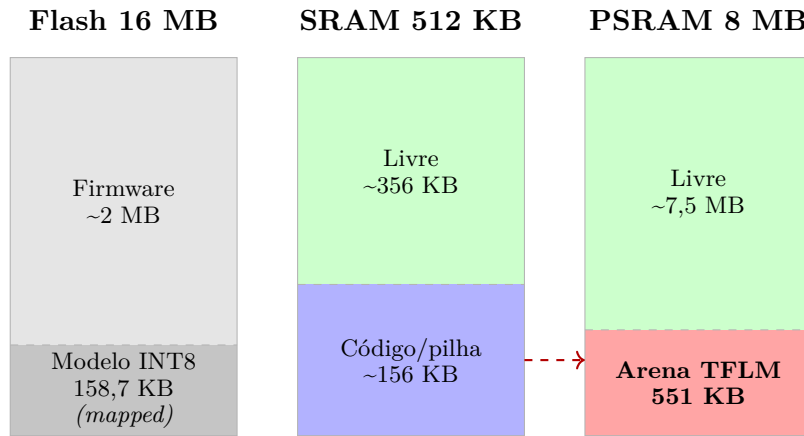
A Figura 9 ilustra a distribuição dos componentes pelos três bancos de memória.

4.3.2 Latência e Determinismo

O jitter de 1 ms em 492 execuções consecutivas indica comportamento determinístico, adequado para sistemas com requisitos de repetibilidade. A latência de 6,1 s é aceitável no contexto de missão orbital, onde a cadência de captura é tipicamente superior a 10 s.

A diferença de $1.000\times$ em relação ao STM32N6 com NPU Ethos-U55 reportada por (Capogrosso; Magno) (0,68–6,45 mJ, 3–30 ms) (Capogrosso; Magno, 2026) é explicada

Figura 9 – Distribuição de memória no ESP32-S3-WROOM-1 durante inferência (alturas ilustrativas)



Arena (551 KB) excede a SRAM disponível (≈ 256 KB)

Alturas não estão em escala real

Fonte: Elaborado pelo autor.

Tabela 15 – Latência de inferência — ESP32-S3, run 20260520_151533 (n = 492)

Métrica	Valor	Observação
Média	6103 ms	—
Mínimo	6103 ms	—
Máximo	6104 ms	Jitter de 1 ms
FPS real	0,164	—

Fonte: Elaborado pelo autor.

por: (1) ausência de NPU no ESP32-S3; (2) penalidade da PSRAM (10–20× menor largura de banda que SRAM interna); (3) ausência de *pruning* estruturado no modelo.

4.4 POLÍTICA DE DECISÃO DE MISSÃO

4.4.1 Comparativo de Estratégias

Foram simuladas quatro famílias de estratégias (S0–S4) sobre os 492 scores de hardware. A Tabela 16 apresenta os principais resultados.

4.4.2 Equivalência S3 vs. S0 em Hardware INT8

A simulação em scores float32 previa $FN = 33$ para S3 vs. $FN = 35$ para S0 argmax, um benefício de apenas 2 imagens. No hardware INT8 real, ambas as estratégias produzem $FN = 35$. A quantização INT8 desloca os scores das 2 imagens borderline (confiança

Tabela 16 – Comparativo de estratégias de decisão — hardware real ($n = 492$)

Estratégia	Recall	Spec.	FN	FP	Vol.	MScore
Transmite tudo (sem modelo)	1,0000	0,0000	0	123	100%	0,6125
S0 Argmax (baseline)	0,9051	0,5854	35	51	78,2%	0,7876
S3 Dual (Tn=Tc=0,40) — adotado	0,9051	0,5854	35	51	78,2%	0,7876
S2 T=0,30 (max MScore)	0,7425	0,9106	95	11	57,9%	0,8342

Fonte: Elaborado pelo autor.

Nota: $\text{MScore} = 0,50 \times \text{Recall} + 0,35 \times \text{Spec.} + 0,15 \times \text{Prec.}$

$\approx 0,40$ em float32) para valores que ainda excedem ambos os limiares ($\text{conf} \geq 0,40 \wedge \text{P}(\text{nuvem}) \geq 0,40$), mantendo o descarte.

Esta discrepância entre simulação e hardware é metodologicamente importante: validações de política de decisão baseadas em scores float32 podem superestimar marginalmente o benefício de limiarização sobre hardware INT8, especialmente para casos borderline próximos dos limiares.

S3 é mantido no firmware por **interpretabilidade**: implementa explicitamente a política mandatória "quando incerto, transmite" com custo computacional zero (3 comparações após a inferência de 6,1 s).

4.5 ANÁLISE ENERGÉTICA

4.5.1 Energia de Transmissão

Com base nos dados de potência do sistema BIRDS-3 (Mehrijardi *et al.*, 2017) e na revisão de comunicações CubeSat de (Saeed *et al.*) (Saeed *et al.*, 2019):

$$E_{\text{TX}}(4800 \text{ bps}) = 98.304 \text{ bits} \times 0,95 \text{ mJ/bit} \approx \mathbf{93,4 \text{ J}} \quad (4.1)$$

O valor de 98.304 bits corresponde a uma imagem de 64×64 px em RGB ($64 \times 64 \times 3 \times 8$ bits), a mesma resolução de entrada do classificador. Essa é uma estimativa conservadora: a resolução de 64×64 é adotada apenas como requisito da rede neural para a decisão de transmitir/descartar, e não necessariamente a resolução nativa de captura do sensor de imagem embarcado. Caso a câmera do nanosatélite capture e transmita imagens em resolução maior — avaliando-se apenas uma versão reduzida (*thumbnail*) de 64×64 para a inferência —, a energia de transmissão poupada por imagem descartada cresce proporcionalmente ao tamanho real do payload, tornando o saldo energético do sistema ainda mais favorável do que o estimado nesta análise.

4.5.2 Energia de Inferência (Estimada)

Com base no datasheet do ESP32-S3 (Espressif Systems, 2024), assumindo corrente de 400 mA a 3,3 V durante operação ativa (CPU 240 MHz + PSRAM):

$$E_{\text{inf}} = 1,32 \text{ W} \times 6,103 \text{ s} \approx 8,1 \text{ J} \quad (\pm 30\%) \quad (4.2)$$

Nota: estimativa baseada em datasheet; medição direta requer instrumento de aferição de corrente (ex.: INA219), identificada como trabalho futuro.

Adicionalmente, essa estimativa é conservadora quanto ao consumo real. O valor de corrente do datasheet (Espressif Systems, 2024) reflete a operação típica do ESP32-S3 com os periféricos de rádio (Wi-Fi/Bluetooth) habilitados, e é amplamente documentado que há uma diferença significativa de consumo de corrente entre os módulos de rádio ligados e desligados. Como a inferência do classificador é uma tarefa puramente computacional (CPU + PSRAM), sem uso de rádio, desabilitar Wi-Fi/Bluetooth durante a etapa de classificação tende a reduzir ainda mais a corrente ativa medida, o que tornaria E_{inf} menor que os 8,1 J estimados e o sistema, consequentemente, ainda mais viável do ponto de vista energético. A confirmação desse efeito depende da medição direta mencionada acima.

4.5.3 Análise de Break-even

A taxa de descarte mínima para equilíbrio energético é:

$$R^* = \frac{E_{\text{inf}}}{E_{\text{TX}}} = \frac{8,1}{46,7} \approx 17,3\% \quad (\text{UHF } 9.600 \text{ bps}) \quad (4.3)$$

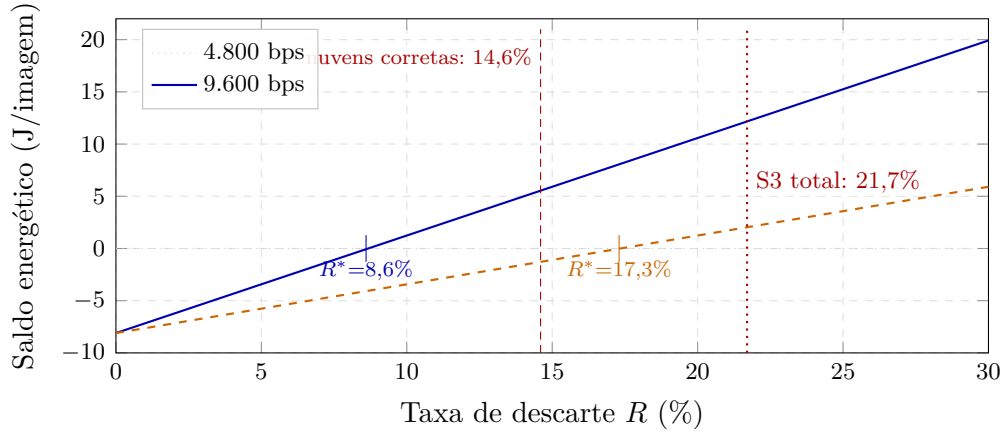
Com taxa de descarte total observada de 21,7% (107/492), o saldo energético estrito é positivo nos dois cenários UHF avaliados: +6,01 kJ a 4.800 bps e +1,01 kJ a 9.600 bps para 492 imagens. Em leitura conservadora, contando apenas nuvens corretamente filtradas (72/492 = 14,6%), o saldo permanece positivo a 4.800 bps (+2,74 kJ), mas torna-se negativo a 9.600 bps (−0,62 kJ). Portanto, em links mais rápidos, o valor primário do sistema é também operacional (redução do orçamento de downlink e priorização de dados), não apenas energético.

A Figura 10 representa graficamente o saldo energético líquido para os dois cenários de taxa de downlink, em função da taxa de descarte.

4.5.4 Valor Operacional Além da Energia

A análise de break-even demonstra que a justificativa energética é função da taxa de downlink. Em geral, o classificador tende a produzir valor operacional complementar ao ganho energético:

Figura 10 – Saldo energético líquido por imagem em função da taxa de descarte



Fonte: Elaborado pelo autor.

Nota: Saldo positivo (>0): o sistema poupa mais energia do que consome ao classificar. R^* : taxa mínima de descarte para equilíbrio. Linhas vermelhas: descarte total S3 e subconjunto conservador de nuvens corretamente descartadas.

- **Orçamento de downlink:** a janela de visibilidade em LEO é de 5–15 min por passagem (Saeed *et al.*, 2019). A UHF 9.600 bps, uma passagem de 10 min permite transmitir $\lfloor \frac{10 \times 60 \times 9600}{98304} \rfloor \approx 58$ imagens de 64×64 RGB. O descarte total de 21,7% (S3 Dual Threshold, 107/492 imagens) libera o equivalente a ≈ 13 imagens por passagem, ao custo de 35 descartes indevidos de dados úteis no conjunto avaliado;
- **Qualidade dos dados recebidos:** imagens de nuvem descartadas não chegam à estação terrestre, eliminando carga de armazenamento, processamento e triagem manual que seria necessária no solo. Em missões com equipe enxuta, essa redução de carga operacional tem impacto prático relevante;
- **Autonomia em cenários de enlace degradado:** em passagens com qualidade de sinal reduzida ou janela de visibilidade encurtada, a triagem embarcada favorece que os dados transmitidos sejam de maior valor científico, benefício não capturado pela análise energética pura e condicionado à acurácia do classificador (82,1% no conjunto avaliado).

4.6 POSICIONAMENTO EM RELAÇÃO AO ESTADO DA ARTE

A Tabela 17 posiciona o presente trabalho em relação a sistemas análogos de inferência embarcada em nanosatélites. Comparações diretas de acurácia são limitadas pela heterogeneidade de tarefas (binária vs. multiclasse) e datasets. As métricas de latência e energia são mais diretamente comparáveis.

O presente trabalho se distingue por reportar validação end-to-end em hardware real com conjunto multiclasse balanceado e análise de decisão formal. A latência de 6.103 ms

Tabela 17 – Comparativo com sistemas análogos de inferência embarcada em nanosatélites

Trabalho	Hardware	Tarefa	Acurácia	Latência	Energia/inf.
Este TCC	ESP32-S3 + PS-RAM 8 MB	4 classes EO	0,821 BAcc	6.103 ms	≈ 8 J (est.)
CubeSatNet (Maskey; Cho, 2020)	ARM Cortex MCU	Binário nuvem	$\approx 90\%$	N/D	N/D
TinyML CubeSat (Capogrosso; Magno, 2026)	STM32N6 + NPU	Multiclasse EO	variável	3–30 ms	0,68–6,45 mJ
VERTECS (Chatar <i>et al.</i> , 2024)	RPi CM4	Binário estelar	$\approx 100\%$	N/D	N/D
CloudScout (Giuffrida <i>et al.</i> , 2020)	PhiSat-1 FPGA	Binário nuvem	$> 90\%$	N/D	N/D
Alainsat-1 (Cabeza-Leiva <i>et al.</i> , 2022)	MCU dedicado	Binário nuvem	N/D	N/D	N/D

Fonte: Elaborado pelo autor com base nas referências citadas.

Nota: N/D: não divulgado no artigo. BAcc: balanced accuracy. EO: Earth Observation. Latência e energia reportadas pelo sistema de origem; metodologias de medição heterogêneas.

e a energia estimada de ≈ 8 J situam o ESP32-S3 como duas a três ordens de grandeza acima do estado da arte com NPU em termos de eficiência (v. Seção 4.3). Três diferenças metodológicas distinguem esta contribuição dos trabalhos análogos:

1. **Validação direta no hardware INT8:** os 492 scores usados na análise da política S3 são provenientes de inferências reais no hardware, não de simulações float32 ou emuladores. A divergência $S3 \approx S0$ em INT8 (vs. $FN = 33$ em float32) só é detectável com validação direta no alvo, reforçando o princípio central do *MLPerf Tiny Benchmark* (Banbury *et al.*, 2021);
2. **Dataset documentado e reprodutível:** o benchmark foi construído por curadoria automática, com critérios explícitos de seleção, remapeamento e divisão ($seed = 42$), seguindo boas práticas de (Geburu *et al.*) (Geburu *et al.*, 2021). Os manifestos `benchmark-v1.csv` e `splits-v1.csv` permitem rastrear as amostras, reproduzir a composição e expandir a curadoria em versões futuras;
3. **Política de decisão com fundamento formal:** a derivação da política S3 a partir de (Elkan) (Elkan, 2001), com limiar ótimo $\theta^* = C(FP)/[C(FP) + C(FN)]$, contrasta com sistemas que adotam limiares de confiança *ad hoc* sem justificativa de custo assimétrico.

5 CONCLUSÃO

5.1 SÍNTESE DOS RESULTADOS

Este trabalho demonstrou a viabilidade de um sistema de classificação embarcada de imagens de observação da Terra em microcontrolador de baixo custo (ESP32-S3), capaz de apoiar a decisão automática de transmitir ou descartar imagens antes do downlink em nanosatélites. Os resultados obtidos constituem uma **prova de conceito**: o componente de classificação foi validado sobre imagens de datasets públicos em hardware real, e a decisão de downlink efetiva dependeria, em uma missão operacional, de fatores complementares como posição orbital, estado do ADCS, orçamento de energia da passagem e objetivos específicos da missão.

Os principais resultados obtidos foram:

- **Acurácia:** o modelo *cnn_v2_64* atingiu acurácia balanceada de 82,1% no conjunto de teste (492 amostras, hardware real), com zero degradação pós-quantização INT8 e comportamento determinístico (jitter de 1 ms em 492 inferências);
- **Embarcabilidade:** o modelo operou com sucesso no ESP32-S3-WROOM-1-N16R8, usando arena TFLM de 551 KB alocada em PSRAM, dentro dos 8 MB disponíveis;
- **Latência:** 6,1 s por inferência, aceitável para a cadência de captura orbital ($\gg 10$ s), mas incompatível com processamento em tempo real; o sistema opera no paradigma de decisão pós-captura;
- **Política de decisão:** *S3 Dual Threshold* (descartar somente se $\text{pred} = \text{nuvem} \wedge \text{conf} \geq 0,40 \wedge P(\text{nuvem}) \geq 0,40$) implementa formalmente a política custo-assimétrica de (Elkan) (Elkan, 2001), alcançando Recall de 90,5% com 21,7% de economia de downlink;
- **Energia:** estimativa de 8 J por inferência ($\pm 30\%$) vs. 47–93 J por imagem transmitida, com saldo energético positivo em links UHF de 4.800 bps e resultado dependente da leitura de descarte em 9.600 bps.

5.2 LIMITAÇÕES

As principais limitações do trabalho são:

1. **Energia não medida diretamente:** a estimativa de 8 J $\pm 30\%$ é baseada em corrente do datasheet (Espressif Systems, 2024), sem medição direta de corrente no circuito.

- Medição precisa requer instrumento de aferição (ex.: INA219 com amostragem de alta resolução);
2. **Dataset desacoplado do hardware final e curadoria automática:** o benchmark foi construído com EuroSAT (Helber *et al.*, 2019) e CloudSEN12 (Aybar *et al.*, 2022), que emulam o domínio orbital mas não são imagens reais do SLCam/UFSC. As amostras não foram modificadas manualmente; os rótulos seguem as fontes públicas e regras de remapeamento documentadas. A generalização para imagens reais da missão requer validação futura;
 3. **Sobreposição espectral intrínseca mista $\leftrightarrow$ nuvem** (documentada na subseção 4.1.3): nenhuma estratégia de *threshold* elimina a sobreposição entre essas classes em 64×64 RGB, resultando em 35 falsos negativos (8,9% do dado útil);
 4. **Integração SLCam não realizada:** o sistema foi validado com protocolo UART desacoplado; a integração com o hardware de imageamento real da missão está fora do escopo desta iteração;
 5. **Escopo restrito ao componente de classificação:** o trabalho valida a viabilidade do módulo de inferência embarcada, e não um sistema completo de apoio à decisão de downlink. A integração com demais subsistemas da missão (ADCS, GNSS/OBC, política de captura, orçamento de energia da passagem) constitui requisito para aplicação operacional e está fora do escopo desta prova de conceito;
 6. **Imagens de datasets públicos, não imagens reais da missão:** o benchmark foi construído a partir do EuroSAT (Helber *et al.*, 2019) e CloudSEN12 (Aybar *et al.*, 2022), que representam o domínio de observação da Terra em condições favoráveis (imagens bem enquadradas, focadas, orientadas para a Terra). Não foram incluídas imagens com degradações típicas de voo — desfoque por movimento, imagens do horizonte ou do céu por falha de apontamento, nem variações de iluminação extremas (eclipse, ângulo solar rasante). A generalização do modelo para essas condições requer validação com dados reais de missão;
 7. **Limiares de decisão definidos empiricamente:** os limiares de confiança da política S3 ($T_n = T_c = 0,40$) foram determinados por análise exploratória sobre o conjunto de teste, sem otimização formal com base em dados operacionais reais de missão nem em requisitos quantificados de falsos negativos aceitáveis estabelecidos pela equipe de missão;
 8. **Hardware COTS sem qualificação espacial:** o ESP32-S3 é um componente comercial (*commercial-off-the-shelf*, COTS) desenvolvido para aplicações terrestres. O uso em missão CubeSat real demandaria avaliação de tolerância a radiação

(partículas energéticas, SEU, *Single Event Upsets*), mecanismos de recuperação de falhas (*watchdog*, reinicialização autônoma), ciclagem térmica em ambiente de vácuo e conformidade com os requisitos de confiabilidade da plataforma. A escolha do ESP32-S3 neste trabalho foi motivada pela validação da metodologia em plataforma acessível, e não pela adequação direta ao ambiente espacial.

5.3 PRÉ-REQUISITOS PARA IMPLANTAÇÃO OPERACIONAL

A transição do sistema do contexto de prova de conceito para uma missão CubeSat real requer que a plataforma espacial satisfaça um conjunto de condições que não foram impostas, nem avaliadas, no presente trabalho. Destacam-se:

1. **Controle de atitude (ADCS):** o modelo foi treinado exclusivamente com imagens da superfície terrestre bem enquadradas. Em CubeSats sem controle de atitude adequado, situação comum em missões universitárias de baixo custo, a câmera pode registrar o céu, o horizonte ou imagens completamente fora de foco, condições não representadas no conjunto de treinamento. O classificador pressupõe que a câmera esteja apontada para a Terra no momento da captura;
2. **Estabilidade angular compatível com o tempo de exposição:** manobras de atitude ou instabilidades residuais durante o tempo de exposição podem produzir borramento por movimento (*motion blur*), alterando as características espectrais e espaciais que o modelo aprendeu a reconhecer em imagens estáticas;
3. **Ausência de manobras durante a captura:** o ciclo captura-classifica-decide pressupõe que o satélite não esteja executando manobras que perturbem a câmera no instante de captura;
4. **Foco previamente calibrado e mantido:** o sistema não inclui mecanismo de avaliação ou correção de foco. Assume-se que o foco está ajustado para a altitude orbital de operação e se mantém estável ao longo da missão;
5. **Condições de iluminação compatíveis com o conjunto de treinamento:** o benchmark foi composto por imagens de satélite diurnas em condições normais de iluminação. Imagens capturadas próximo ao terminador (alba/crepúsculo orbital) ou sob ângulos solares extremos podem apresentar características fora da distribuição de treinamento;
6. **Conhecimento da posição orbital:** se o objetivo da missão for selecionar apenas imagens sobre uma região geográfica de interesse (p.ex., passagem sobre o Brasil), o sistema precisará conhecer sua posição orbital no momento da captura. Em

implementação real, isso requer módulo GNSS embarcado ou propagador orbital com elementos TLE atualizados pelo comando de solo;

7. **Análise da cadência de captura e impacto no power budget:** o impacto energético real do sistema depende diretamente da frequência de captura de imagens ao longo da órbita. Uma análise detalhada da cadência de captura, combinada com medições diretas de consumo durante a inferência, é necessária para avaliar o efeito real sobre o orçamento de energia da missão;
8. **Alternativas de hardware com maior adequação ao ambiente espacial:** existem arquiteturas de processamento computacionalmente equivalentes ao ESP32-S3 com maior adequação ao ambiente espacial, tanto em robustez a radiação quanto em consumo de energia. Plataformas com NPU dedicada (como o STM32N6 ou equivalentes com suporte a normas ECSS) reduziriam a latência e a energia por inferência em duas a três ordens de grandeza (Capogrosso; Magno, 2026). A decisão por utilizar o ESP32-S3 neste trabalho foi motivada pela validação da metodologia com hardware acessível e de baixo custo, e não pela intenção de utilizar esse processador diretamente em uma missão espacial.

5.4 TRABALHOS FUTUROS

Com base nas limitações identificadas e nas oportunidades abertas pelos resultados, propõe-se:

1. **Medição direta de energia:** instrumentar o ESP32-S3 com sensor de corrente (INA219 ou shunt de precisão + osciloscópio) para substituir a estimativa de $\pm 30\%$ por dado empírico;
2. **Pruning estruturado do modelo:** aplicar esparsidade de 70–90% ao *cnn_v2_64* e requalificar, reduzindo arena e latência em 3–5× para aproximar o sistema do estado da arte em eficiência energética reportado por (Capogrosso; Magno) (Capogrosso; Magno, 2026);
3. **Integração com SLCam:** adaptar o protocolo UART para o hardware de imageamento real da missão SpaceLab/UFSC;
4. **Dataset com imagens reais e curadoria estendida:** expandir o benchmark com imagens reais do SLCam ou de missões análogas, incluindo revisão de casos limítrofes e deduplicação visual, para validar a generalização do classificador;
5. **Hardware com NPU:** avaliar plataformas com NPU dedicado (STM32N6, i.MXR-T117x) para quantificar o potencial de redução de latência e energia identificado na literatura (Capogrosso; Magno, 2026);

6. **Limiar de confiança por classe:** explorar thresholds distintos para *mista* (mais restrito) vs. *nuvem* (menos restrito) para compensar a assimetria de confusão identificada.

5.5 CONSIDERAÇÕES FINAIS

O trabalho posicionou-se na interseção entre dois campos ativos: classificação de cena em imagens de observação da Terra (Lima; Marfurt, 2020) e TinyML em microcontroladores (Lin *et al.*, 2020; Banbury *et al.*, 2021). A combinação de benchmark reprodutível e documentado, análise de decisão com fundamentação formal e prova de conceito em hardware real posiciona este trabalho como ponto de partida para desenvolvimentos de maior maturidade tecnológica.

A equivalência entre a política S3 *Dual Threshold* e o argmax simples no hardware INT8 é um resultado honesto e metodologicamente relevante: demonstra que simulações baseadas em scores float32 podem superestimar benefícios de limiarização em hardware quantizado, reforçando o princípio central do *MLPerf Tiny Benchmark* (Banbury *et al.*, 2021): validar sempre diretamente no hardware alvo.

O trabalho demonstra que a fronteira entre pesquisa acadêmica em sistemas embarcados e aplicações espaciais reais pode ser explorada com plataformas acessíveis, desde que as limitações sejam declaradas com clareza. Com hardware de prateleira (ESP32-S3), dados públicos (EuroSAT (Helber *et al.*, 2019), CloudSEN12 (Aybar *et al.*, 2022)) e infraestrutura de software aberta (TensorFlow Lite for Microcontrollers, ESP-IDF), foi possível construir um sistema de classificação de imagens de observação da Terra com acurácia balanceada de 82,1% e operação demonstrada em hardware real. A contribuição central não reside na acurácia absoluta, pois sistemas com NPU dedicado alcançam latência e energia duas a três ordens de grandeza melhores, mas na **metodologia**: estruturar o problema de missão em termos de decisão custo-assimétrica, escolher métricas adequadas ao contexto orbital, construir um benchmark documentável e validar o sistema completo diretamente no hardware alvo. Esses princípios metodológicos são transferíveis a plataformas de maior maturidade tecnológica, e é nessa transferibilidade que reside o valor acadêmico deste trabalho. Importante ressaltar, contudo, que os resultados não constituem uma validação em ambiente operacional real: essa etapa requer dados reais de missão, medições diretas de energia e a integração com os demais subsistemas da plataforma espacial descritos na Seção 5.3.

REFERÊNCIAS

AYBAR, Cesar *et al.* CloudSEN12, a global dataset for semantic understanding of cloud and cloud shadow in Sentinel-2. **Scientific Data**, Nature Publishing Group, v. 9, p. 582, 2022. DOI: [10.1038/s41597-022-01878-2](https://doi.org/10.1038/s41597-022-01878-2). Disponível em: <https://doi.org/10.1038/s41597-022-01878-2>. Citado nas pp. 17 sq., 20, 27 sq., 38 sq., 60, 63, 76.

BANBURY, Colby *et al.* **MLPerf Tiny Benchmark**. [S. l.: s. n.], 2021. arXiv: [2106.07597](https://arxiv.org/abs/2106.07597) [cs.LG]. Disponível em: <https://arxiv.org/abs/2106.07597>. Citado nas pp. 23, 32 sqq., 46, 58, 63.

BROCKMANN, Markus; SCHLIPPE, Tim. Optimizing Convolutional Neural Networks for Image Classification on Resource-Constrained Microcontroller Units. **Computers**, MDPI, v. 13, n. 7, p. 173, 2024. DOI: [10.3390/computers13070173](https://doi.org/10.3390/computers13070173). Disponível em: <https://doi.org/10.3390/computers13070173>. Citado nas pp. 23, 43.

CABEZA-LEIVA, Antonio *et al.* **On-board image classification payload for a 3U CubeSat using machine learning for on-orbit cloud detection**. [S. l.], 2022. Projeto Alainsat-1 — detecção de nuvens em nanosatélite. UCommons Repositori Institucional. Disponível em: <https://upcommons.upc.edu/handle/2117/370867>. Citado nas pp. 30, 58.

CAPOGROSSO, Luigi; MAGNO, Michele. **TinyML Enhances CubeSat Mission Capabilities: Efficient On-Board Inference for Earth Observation**. [S. l.: s. n.], 2026. Preprint aceito para ICCPS 2026 (ACM/IEEE). Disponível em: arXiv:2603.20174. arXiv: [2603.20174](https://arxiv.org/abs/2603.20174) [cs.LG]. Disponível em: <https://arxiv.org/abs/2603.20174>. Citado nas pp. 15 sq., 23 sq., 30, 35, 53, 58, 62.

CHATAR, Keenan A. *et al.* **Data downlink prioritization using image classification on-board a 6U CubeSat**. [S. l.: s. n.], 2024. SPIE Remote Sensing / arXiv. Experimento VERTECS — Kyushu Institute of Technology. arXiv: [2408.14865](https://arxiv.org/abs/2408.14865) [cs.CV]. Disponível em: <https://arxiv.org/abs/2408.14865>. Citado nas pp. 15, 22, 30, 58.

ELKAN, Charles. The Foundations of Cost-Sensitive Learning. In: PROCEEDINGS of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI). [S. l.]: Morgan Kaufmann, 2001. v. 17, p. 973–978. Disponível em: <https://www.ijcai.org/Proceedings/01/IJCAI-2001.pdf>. Citado nas pp. 16, 18, 20, 30 sqq., 47, 58 sq.

ESPRESSIF SYSTEMS. **ESP32-S3 Series Datasheet**. v2.2. ed. [S. l.], 2024. Disponível em: https://www.espressif.com/sites/default/files/documentation/esp32-s3_datasheet_en.pdf. Citado nas pp. 24, 34 sq., 45, 56, 59.

ESPRESSIF SYSTEMS. **ESP32-S3-WROOM-1 & ESP32-S3-WROOM-1U Datasheet**. v1.6. ed. [S. l.], 2023. Disponível em:

https://www.espressif.com/sites/default/files/documentation/esp32-s3-wroom-1_wroom-1u_datasheet_en.pdf. Citado na p. 45.

EUROPEAN SPACE AGENCY (ESA). **Sentinel-2 User Handbook**. [S. l.], 2022. ESA Standard Document. Revisão 1. Disponível em: https://sentinel.esa.int/documents/247904/685211/Sentinel-2_User_Handbook. Citado na p. 27.

GEBRU, Timnit *et al.* Datasheets for Datasets. **Communications of the ACM**, v. 64, n. 12, p. 86–92, 2021. DOI: [10.1145/3458723](https://doi.org/10.1145/3458723). Disponível em: <https://doi.org/10.1145/3458723>. Citado nas pp. 17 sq., 28, 37 sq., 58, 75.

GIUFFRIDA, Gianluca *et al.* CloudScout: A Deep Neural Network for On-Board Cloud Detection on Hyperspectral Images. **Remote Sensing**, MDPI, v. 12, n. 14, p. 2205, 2020. DOI: [10.3390/rs12142205](https://doi.org/10.3390/rs12142205). Disponível em: <https://doi.org/10.3390/rs12142205>. Citado nas pp. 15 sq., 21 sq., 24, 29 sq., 58.

GOOGLE LLC. **TensorFlow Lite for Microcontrollers**. [S. l.: s. n.], 2023. Software framework. Versão 2.16. Acesso em: maio 2026. Disponível em: <https://www.tensorflow.org/lite/microcontrollers>. Citado na p. 37.

HELBBER, Patrick *et al.* EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 12, n. 7, p. 2217–2226, 2019. DOI: [10.1109/JSTARS.2019.2918242](https://doi.org/10.1109/JSTARS.2019.2918242). Disponível em: <https://doi.org/10.1109/JSTARS.2019.2918242>. Citado nas pp. 17 sq., 20, 27 sq., 38, 60, 63, 76.

IOFFE, Sergey; SZEGEDY, Christian. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *In*: INTERNATIONAL Conference on Machine Learning (ICML). [S. l.: s. n.], 2015. v. 37, p. 448–456. Disponível em: <http://proceedings.mlr.press/v37/ioffe15.html>. Citado na p. 25.

JACOB, Benoit *et al.* Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. *In*: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). [S. l.: s. n.], 2018. p. 2704–2713. DOI: [10.1109/CVPR.2018.00286](https://arxiv.org/abs/1712.05877). Disponível em: <https://arxiv.org/abs/1712.05877>. Citado nas pp. 18, 20, 25 sq., 40, 45, 52.

LASSEUR, Christophe *et al.* Review on Hardware Devices and Software Techniques Enabling Neural Network Inference Onboard Satellites. **Remote Sensing**, MDPI, v. 16, n. 21, p. 3957, 2024. DOI: [10.3390/rs16213957](https://doi.org/10.3390/rs16213957). Disponível em: <https://doi.org/10.3390/rs16213957>. Citado nas pp. 14 sq., 24.

LIMA, Rodolfo P. de; MARFURT, Kurt. Convolutional Neural Network for Remote-Sensing Scene Classification: Transfer Learning Analysis. **Remote Sensing**, MDPI, v. 12, n. 1, p. 86, 2020. DOI: [10.3390/rs12010086](https://doi.org/10.3390/rs12010086). Disponível em: <https://doi.org/10.3390/rs12010086>. Citado nas pp. 22, 26, 63.

LIN, Ji *et al.* MCUNet: Tiny Deep Learning on IoT Devices. *In: ADVANCES in Neural Information Processing Systems (NeurIPS)*. [S. l.: s. n.], 2020. v. 33, p. 11711–11722. Disponível em: <https://proceedings.neurips.cc/paper/2020/hash/86c51678350f656dcc7f490a43946ee5-Abstract.html>. Citado nas pp. 16, 22 sq., 63.

MASKEY, Abhas; CHO, Mengu. CubeSatNet: Ultralight Convolutional Neural Network designed for on-orbit binary image classification on a 1U CubeSat. **Engineering Applications of Artificial Intelligence**, Elsevier, v. 96, p. 103952, 2020. CubeSatNet — classificação binária embarcada em nanosatélite universitário 1U. DOI: [10.1016/j.engappai.2020.103952](https://doi.org/10.1016/j.engappai.2020.103952). Disponível em: <https://doi.org/10.1016/j.engappai.2020.103952>. Citado nas pp. 15, 22, 29 sq., 58.

MATEO-GARCIA, Gonzalo *et al.* Towards global flood mapping onboard low cost satellites with machine learning. **Scientific Reports**, Nature Publishing Group, v. 13, p. 1–15, 2023. DOI: [10.1038/s41598-023-34436-w](https://doi.org/10.1038/s41598-023-34436-w). Disponível em: <https://doi.org/10.1038/s41598-023-34436-w>. Citado nas pp. 15, 22, 30.

MEHRJARDI, Mohammad Fereydoon *et al.* Power Budgets for CubeSat Radios to Support Data Downlink. **IEEE Aerospace and Electronic Systems Magazine**, v. 32, n. 6, p. 4–16, 2017. Dados de potência do BIRDS-3: 4,56 W @ 4.800 bps UHF. Disponível em: https://digitalcommons.odu.edu/engtech_fac_pubs/29. Citado nas pp. 14, 18, 34 sqq., 55.

SAEED, Nasir *et al.* **CubeSat Communications: Recent Advances and Future Challenges**. [S. l.: s. n.], 2019. arXiv: [1908.09501](https://arxiv.org/abs/1908.09501) [cs.NI]. Disponível em: <https://arxiv.org/abs/1908.09501>. Citado nas pp. 14, 34 sq., 55, 57.

TRAN, Daniel *et al.* **The Autonomous Sciencecraft Experiment Onboard the EO-1 Spacecraft**. [S. l.], 2004. Disponível em: <https://ntrs.nasa.gov/citations/20060043562>. Citado nas pp. 15, 21 sq.

Apêndices

APÊNDICE A – ARQUITETURA DO MODELO *CNN_V2_64*

O modelo *cnn_v2_64* é uma CNN customizada projetada para embarcabilidade no ESP32-S3. A arquitetura segue o padrão VGG-like com três blocos convolucionais e um *head* com *GlobalAveragePooling2D*, escolhido por compatibilidade com o *TensorFlow Lite for Microcontrollers* (TFLM), que não suporta operações de *Flatten* de tensor arbitrário.

HIPERPARÂMETROS DE TREINAMENTO

Tabela 18 – Hiperparâmetros de treinamento do modelo *cnn_v2_64*

Parâmetro	Valor	Observação
Resolução de entrada	$64 \times 64 \times 3$	RGB, float32 [0, 255]
Número de classes	4	mista, nuvem, oceano, terra
Filtros base	32	dobro a cada bloco
Épocas máximas	80	<i>early stopping</i> aplicado
<i>Early stopping patience</i>	20	monitorando <code>val_accuracy</code>
<i>Reduce LR patience</i>	5	fator 0,1
Taxa de aprendizado	0,001	Adam optimizer
Tamanho do lote	32	
Pesos de classe	balanceados	1,0 para todas as classes
<i>Dropout</i> (blocos 1, 2, 3, <i>head</i>)	0,10 / 0,15 / 0,20 / 0,30	
Semente aleatória	42	reprodutibilidade garantida
<i>Backend</i> Keras	PyTorch	Keras 3.14.1

Fonte: Elaborado pelo autor.

SUMÁRIO DE CAMADAS

As camadas de augmentação (*RandomFlip*, *RandomRotation*, *RandomZoom*, *RandomBrightness*, *RandomContrast*) são internas ao modelo, porém permanecem inativas durante inferência (`training=False`). Camadas de *Dropout* também são inativas em inferência.

Tabela 19 – Camadas do modelo *cnn_v2_64* (inferência, `training=False`)

Camada	Tipo	Forma de Saída	Parâm.	Observação
<i>Entrada e pré-processamento</i>				
<code>image_input</code>	Input	(64, 64, 3)	0	float32 [0, 255]
<code>aug_*</code> ($\times 5$)	Augmentação	(64, 64, 3)	0	inativas em inferência
<code>rescaling</code>	Rescaling(1/255)	(64, 64, 3)	0	normalização interna
<i>Bloco 1 — 32 filtros, saída 32×32</i>				
<code>b1_conv1</code>	Conv2D(32, 3×3, <i>same</i>)	(64, 64, 32)	864	sem <i>bias</i>
<code>b1_bn1</code>	BatchNormalization	(64, 64, 32)	128	64 treináveis
<code>b1_relu1</code>	Activation(relu)	(64, 64, 32)	0	
<code>b1_conv2</code>	Conv2D(32, 3×3, <i>same</i>)	(64, 64, 32)	9216	sem <i>bias</i>
<code>b1_bn2</code>	BatchNormalization	(64, 64, 32)	128	64 treináveis
<code>b1_relu2</code>	Activation(relu)	(64, 64, 32)	0	
<code>b1_pool</code>	MaxPooling2D(2×2)	(32, 32, 32)	0	<i>stride</i> =2
<code>b1_drop</code>	Dropout(0,10)	(32, 32, 32)	0	inativo em inferência
<i>Bloco 2 — 64 filtros, saída 16×16</i>				
<code>b2_conv1</code>	Conv2D(64, 3×3, <i>same</i>)	(32, 32, 64)	18.432	sem <i>bias</i>
<code>b2_bn1</code>	BatchNormalization	(32, 32, 64)	256	128 treináveis
<code>b2_relu1</code>	Activation(relu)	(32, 32, 64)	0	
<code>b2_conv2</code>	Conv2D(64, 3×3, <i>same</i>)	(32, 32, 64)	36.864	sem <i>bias</i>
<code>b2_bn2</code>	BatchNormalization	(32, 32, 64)	256	128 treináveis
<code>b2_relu2</code>	Activation(relu)	(32, 32, 64)	0	
<code>b2_pool</code>	MaxPooling2D(2×2)	(16, 16, 64)	0	<i>stride</i> =2
<code>b2_drop</code>	Dropout(0,15)	(16, 16, 64)	0	inativo em inferência
<i>Bloco 3 — 128 filtros, saída 8×8</i>				
<code>b3_conv1</code>	Conv2D(128, 3×3, <i>same</i>)	(16, 16, 128)	73.728	sem <i>bias</i>
<code>b3_bn1</code>	BatchNormalization	(16, 16, 128)	512	256 treináveis
<code>b3_relu1</code>	Activation(relu)	(16, 16, 128)	0	
<code>b3_pool</code>	MaxPooling2D(2×2)	(8, 8, 128)	0	<i>stride</i> =2
<code>b3_drop</code>	Dropout(0,20)	(8, 8, 128)	0	inativo em inferência
<i>Cabeça classificadora</i>				
<code>gap</code>	GlobalAveragePooling2D	(128,)	0	compatível TFLM
<code>head_dense</code>	Dense(64)	(64,)	8192	sem <i>bias</i>
<code>head_bn</code>	BatchNormalization	(64,)	256	128 treináveis
<code>head_relu</code>	Activation(relu)	(64,)	0	
<code>head_drop</code>	Dropout(0,30)	(64,)	0	inativo em inferência
<code>predictions</code>	Dense(4, <i>softmax</i>)	(4,)	260	com <i>bias</i>
Total treináveis			148.324	
Total não-treináveis			768	médias móveis BN
Total de parâmetros			149.092	

Fonte: Elaborado pelo autor.

Tabela 20 – Pegada de memória do *cnn_v2_64* em diferentes formatos

Formato	Armazenamento	Tamanho (KB)	Observação
Keras (.keras)	Disco	1868,3	inclui overhead HDF5
TFLite float32	Flash (MCU)	582,4	inferência em float32
TFLite INT8 (PTQ)	Flash (MCU)	158,7	artefato embarcado
Arena TFLM (real, PSRAM)	RAM (PSRAM)	551	medido no ESP32-S3 (TCK-006)

Fonte: Elaborado pelo autor.

PEGADA DE MEMÓRIA

A arena TFLM de 551 kB excede a SRAM interna do ESP32-S3 (512 kB), sendo alocada em PSRAM via `heap_caps_malloc(MALLOC_CAP_SPIRAM)`. Os pesos (158,7 kB) são armazenados em Flash e carregados via mapeamento de memória, não consumindo RAM.

APÊNDICE B – PROTOCOLO UART

COMPLETO

O protocolo UART bidirecional implementado no firmware do ESP32-S3 é descrito nas Tabelas 21 e 22. O transporte é baseado em linhas de texto terminadas em CRLF para controle, e transferência binária bruta de 12.288 bytes para o *payload* de imagem.

Tabela 21 – Configuração da interface UART do ESP32-S3

Parâmetro	Valor
Baud rate	115.200 bps
Bits de dados	8
Paridade	Nenhuma
Bits de parada	1
Controle de fluxo	Nenhum
Timeout comando	60 s (após isso ESP32 envia ##ALIVE)
Timeout imagem	15 s para receber os 12.288 bytes
Terminador de linha	\r\n (CRLF)
Prefixo de resposta	##

Fonte: Elaborado pelo autor.

Tabela 22 – Comandos do protocolo UART

Host → ESP32	ESP32 → Host	Descrição
PING\r\n	##PONG \r\n	Verificação de conectividade
INFER\r\n	##READY \r\n	Solicitar inferência; ESP32 aguarda <i>payload</i>
[12.288 bytes RGB bruto]	##RES ,<id>,<nome>,<conf>,<s0>,<s1>,<s2>,<s3>,<ms>,<dec>	Enviar imagem 64×64×3; receber resultado
STATUS\r\n	##STATUS ,<arena_kb>,<sram_free_kb>\r\n	Consultar uso de memória
RESET\r\n	##RESET_OK \r\n	Reinicializar dispositivo
—	##ERROR ,TIMEOUT	Timeout de recepção de <i>payload</i>
—	##ERROR ,INFER_FAILED	Falha na execução da inferência
—	##SERVER_READY	Sinalização de prontidão (após <i>boot</i>)
—	##ALIVE	<i>Heartbeat</i> (60 s sem comando do host)

Fonte: Elaborado pelo autor.

A resposta de inferência (**##RES**) contém os seguintes campos separados por vírgula:

- <id>: índice inteiro da classe predita (0 = mista, 1 = nuvem, 2 = oceano, 3 = terra);
- <nome>: nome textual da classe predita;
- <conf>: *score* softmax da classe vencedora (4 casas decimais);
- <s0> a <s3>: *scores* softmax de todas as classes na ordem [mista, nuvem, oceano, terra] (4 casas decimais cada);
- <ms>: latência de inferência em milissegundos (inteiro, medida com `esp_timer_get_time()`, resolução 1 µs);

- **<dec>**: decisão de missão — **TX** (transmitir) ou **DX** (descartar), segundo a política S3 *Dual Threshold*.

Exemplo de resposta completa:

```
##RES,1,nuvem,0.8523,0.0923,0.8523,0.0412,0.0142,6087,DX
```

O fluxo de interação para uma inferência completa é: (1) host envia **INFER\r\n**; (2) ESP32 responde **##READY**; (3) host transmite os 12.288 bytes do *buffer* RGB; (4) ESP32 processa (6,1 s) e responde com **##RES, . . .**

APÊNDICE C – RESULTADOS DETALHADOS DO BENCHMARK

Este apêndice apresenta as métricas completas dos modelos avaliados no protocolo v2 (ablação de resolução, TCK-004 v2), incluindo resultados pós-quantização INT8 e intervalos de confiança. Todos os modelos foram avaliados no conjunto de teste ($n = 492$ amostras, 123 por classe), com semente aleatória 42.

TABELA COMPARATIVA COMPLETA

Tabela 23 – Métricas de todos os modelos no conjunto de teste ($n = 492$)

Modelo	Res.	bal_acc	F1	F1 _{multa}	F1 _{mevem}	F1 _{oceano}	F1 _{terra}	INT8 (KB)	Viável
cnn_v2_64 ^a	64	0,8211	0,8216	0,6818	0,6293	1,0000	0,9752	158,7	Sim
cnn_v2_96 ^a	96	0,8069	0,8070	0,6320	0,6489	0,9919	0,9551	158,7	Sim
cnn_v2_128 ^a	128	0,8089	0,8092	0,6352	0,6590	0,9879	0,9547	158,7	Não ^a
cnn_custom (v1) ^a	64	0,7846	0,7861	0,5726	0,6212	0,9878	0,9627	158,7	Sim
MobileNetV2	64	0,7002	0,7008	0,5844	0,6008	0,9644	0,8936	~5353 ^b	Não ^b
EfficientNet-Lite0	64	0,7561	0,7511	0,5702	0,5975	0,9426	0,8939	~6388 ^b	Não ^b
Baseline Histograma+LR	64	0,7154	0,7135	0,5966	0,6234	0,8814	0,7527	N/A	N/A

^a Modelo selecionado para embarcamento. ^b Arena TFLM estimada ≈540 KB > PSRAM disponível para ativações.

^c Tamanho **estimado** (export INT8 não concluído); inviável para Flash de 16 MB com ESP-IDF.

^d Todos os variantes *cnn_v2* possuem **idênticos** 158,7 KB INT8 pois compartilham a mesma arquitetura. A resolução de entrada (64/96/128) altera apenas os tensores intermediários (arena em RAM), não os pesos gravados em Flash.

Fonte: Elaborado pelo autor.

RESULTADOS PÓS-QUANTIZAÇÃO INT8

Tabela 24 – Comparação float32 vs. INT8 para modelos candidatos ($n = 492$)

Modelo	bal_acc float32	bal_acc INT8	Degradação (pp)	Concordância (%)
cnn_v2_64	0,8211	0,8211	0,000	99,6
cnn_v2_96	0,8069	0,8028	−0,004	—
cnn_v2_128	0,8089	0,8069	−0,002	—
cnn_custom (v1)	0,7846	0,7825	−0,002	—

Fonte: Elaborado pelo autor.

INTERVALOS DE CONFIANÇA (95%, WALD)

Os intervalos foram calculados pela fórmula de Wald: $\hat{p} \pm 1,96\sqrt{\hat{p}(1 - \hat{p})/n}$, com $n = 492$.

Observações estatísticas:

Tabela 25 – Intervalos de confiança de 95% para bal_acc ($n = 492$)

Modelo	bal_acc	IC_{inf}	IC_{sup}	Amplitude
cnn_v2_64	0,8211	0,787	0,855	0,068
cnn_v2_128	0,8089	0,774	0,844	0,069
cnn_v2_96	0,8069	0,772	0,842	0,070
cnn_custom (v1)	0,7846	0,748	0,821	0,072
MobileNetV2	0,7602	0,723	0,798	0,075
EfficientNet-Lite0	0,7561	0,718	0,794	0,076
Baseline Histograma+LR	0,7154	0,676	0,755	0,080

Fonte: Elaborado pelo autor.

- cnn_v2_64 [0,787; 0,855] vs. cnn_v2_96 [0,772; 0,842]: intervalos sobrepostos — diferença **não** significativa ao nível 95%; escolha entre ambos baseia-se em embarcabilidade;
- cnn_v2_64 [0,787; 0,855] vs. Baseline [0,676; 0,755]: intervalos sem sobreposição — superação da *baseline* confirmada estatisticamente;
- cnn_v2_64 vs. cnn_custom [0,748; 0,821]: sobreposição estreita — diferença prática relevante (+3,6 pp) mas *borderline* para significância formal.

ESTRATÉGIAS DE DECISÃO: COMPARATIVO

A Tabela 26 resume as melhores variantes de cada família de estratégia de decisão, avaliadas com os *scores* float32 do cnn_v2_64 em simulação (492 amostras). Recall refere-se a imagens **úteis** corretamente transmitidas; Especificidade refere-se a imagens **nuvem** corretamente descartadas.

Tabela 26 – Melhores variantes de cada estratégia de decisão (simulação float32, $n = 492$)

ID	Estratégia (melhor variante)	TP	FN	FP	Recall	Espec.	MScore
S0	Argmax simples (<i>baseline</i>)	334	35	51	0,905	0,585	—
S1	$Conf \geq 0,40$ para descartar	336	33	51	0,911	0,585	0,790
S2	$P(\text{nuvem}) \leq 0,30 \Rightarrow \text{TX}$	274	95	11	0,742	0,911	0,834
S3	DX se pred=nuvem $\wedge$ conf $\geq 0,40 \wedge P(\text{nuvem}) \geq 0,40$ (recomendada)	336	33	51	0,911	0,585	0,790
S4	Margem conf $\geq 0,05$	339	30	55	0,919	0,553	0,782
S3 real	(hardware INT8, run definitivo)	334	35	51	0,905	0,585	—

MScore = $0,50 \times \text{Recall} + 0,35 \times \text{Espec.} + 0,15 \times \text{Prec.}$, onde $\text{Prec.} = \text{TX}/(\text{TX} + \text{FP})$.S3 real = S0 no hardware INT8 (quantização desloca *scores* *borderline*, tornando S3 equivalente ao argmax).

Fonte: Elaborado pelo autor.

APÊNDICE D – MANIFESTO DO DATASET BENCHMARK-V1

O *benchmark-v1* é o conjunto de dados curado automaticamente utilizado neste trabalho para treinamento, validação e teste dos modelos de classificação, documentado conforme *Datasheets for Datasets* (Gebru *et al.*, 2021). Não houve modificação manual de amostras individuais; a composição resulta de regras de seleção, remapeamento e particionamento reproduzíveis.

IDENTIFICAÇÃO

Tabela 27 – Identificação e propriedades gerais do *benchmark-v1*

Campo	Valor
Identificador	<code>benchmark-v1</code>
Arquivo manifesto	<code>datasets/manifests/benchmark-v1.csv</code>
Total de amostras	3264
Número de classes	4 (mista, nuvem, oceano, terra)
Balanceamento	Balanceado por construção (816 amostras/classe)
Resolução curada	64×64, 96×96 e 128×128 pixels (RGB)
Formato	PNG, 3 canais, uint8 [0, 255]
Semente de divisão	42 (divisão estratificada por classe)
Versão	1 (2026-05-14)

Fonte: Elaborado pelo autor.

COMPOSIÇÃO POR FONTE

Tabela 28 – Distribuição de amostras por fonte e classe no *benchmark-v1*

Fonte	mista	nuvem	oceano	terra	Total
EuroSAT RGB	0	0	816	768	1584
CloudSEN12	816	816	0	48	1680
Total	816	816	816	816	3264

Fonte: Elaborado pelo autor.

Notas sobre as fontes:

- **EuroSAT RGB** (Helber *et al.*, 2019): imagens Sentinel-2 de 64×64 pixels RGB. *AnnualCrop*, *Forest*, *HerbaceousVegetation*, *Pasture* e *PermanentCrop* → **terra**; *SeaLake* → **oceano**. Excluídas: *Highway*, *Industrial*, *Residential*, *River* (sem correspondência na missão);
- **CloudSEN12** (Aybar *et al.*, 2022): anotações de cena em imagens Sentinel-2. A classe **oceano** produziu apenas 2 amostras (dataset enviesado para cenas terrestres), suprida integralmente pelo EuroSAT SeaLake.

DIVISÃO TREINO / VALIDAÇÃO / TESTE

Tabela 29 – Distribuição de amostras por *split* e classe no *benchmark-v1*

<i>Split</i>	mista	nuvem	oceano	terra	Total
Treino	571	571	571	571	2284
Validação	122	122	122	122	488
Teste	123	123	123	123	492
Total	816	816	816	816	3264

Fonte: Elaborado pelo autor.

ESQUEMA DO ARQUIVO MANIFESTO

O arquivo `benchmark-v1.csv` contém os seguintes campos:

- **sample_id**: identificador único no formato BV1-NNNNNN;
- **source_dataset**: fonte de origem (EuroSAT RGB ou CloudSEN12);
- **source_image_id**: identificador da imagem original na fonte;
- **scene_group_id**: agrupamento de cenas para deduplicação (reservado para iteração futura);
- **class_label**: rótulo de classe (**mista**, **nuvem**, **oceano**, **terra**);
- **split**: divisão (**train**, **val**, **test**);
- **crop_path**: caminho relativo do arquivo PNG na resolução 128×128 ;
- **original_width**, **original_height**: dimensões originais;
- **processed_width**, **processed_height**: dimensões após redimensionamento;
- **quality_flags**: sinalizadores de qualidade (vazio = aprovado);
- **review_status**: **auto** (aprovação automática por critérios de composição) ou **manual**;
- **notes**: observações adicionais (e.g., classe original da fonte).

Como limitação, o campo **scene_group_id** foi reservado para iteração futura, sem agrupamento visual de cenas: amostras homogêneas, casos limítrofes **mista**/**nuvem** e duplicatas visuais são preservados na curadoria automática original. Isso evita intervenção manual e mantém a reprodutibilidade, mas versões futuras devem adicionar deduplicação visual e revisão de casos ambíguos.