GEOAI - A TRANSFORMAÇÃO DO GEOPROCESSAMENTO E ESTUDOS AMBIENTAIS ATRAVÉS DE FERRAMENTAS DE INTELIGENCIA ARTIFICIAL

Antônio Edmilson de Alcântara Dantas [1] ; Felipe Daniel de Oliveira Santos [2]; Abner Lins Silva [3] & Davi Diniz de Carvalho Melo [4] (Times New Roman 12 pt, itálico, separados com; e centralizado)

RESUMO – Este trabalho avaliou o desempenho técnico do plugin gratuito GeoAI para QGIS em análises multimodais, segmentação orientada por texto e segmentação semântica de imagens aéreas. O estudo utilizou ortomosaicos do Centro de Tecnologia da Universidade Federal da Paraíba e de uma área do Rio Jaguaribe, em João Pessoa. A instalação foi realizada pelo Pixi, com suporte a GPU e CUDA, sendo documentados problemas de cache, acesso ao SAM 3, resolução de dependências e incompatibilidade do Moondream com a versão 5.12.0 do Transformers. Após as correções, foram testados os modos Caption, Query e Detect do Moondream, além do SamGeo3 e de uma U-Net com codificador ResNet-34. O Moondream apresentou descrição coerente da paisagem, identificou duas rotatórias e detectou quatro conjuntos de painéis solares. O SamGeo3 segmentou satisfatoriamente o canal do Rio Jaguaribe, porém mostrou maior sobreposição na vegetação. Na segmentação semântica, a correção para três classes permitiu distinguir fundo, edificações e árvores. O melhor IoU foi 0,7136, com F1 de 0,7490 e precisão de 0,8975. A inferência processou 3.465 janelas e gerou resultados raster e vetoriais integrados ao ambiente QGIS. Conclui-se que o GeoAI é uma alternativa acessível e promissora, embora exija controle das versões, validação visual e ampliação das amostras.

ABSTRACT– This study evaluated the technical performance of the GeoAI plugin for QGIS in multimodal analysis, text-guided segmentation, and semantic segmentation of aerial imagery. The study used orthomosaics of the Technology Center at the Federal University of Paraíba and the Jaguaribe River area in João Pessoa. The installation was carried out using Pixi, with GPU and CUDA support, and issues related to cache corruption, SAM 3 access, dependency resolution, and incompatibility between Moondream and Transformers version 5.12.0 were documented. After these issues were resolved, the Caption, Query, and Detect modes of Moondream were tested, together with SamGeo3 and a U-Net architecture using a ResNet-34 encoder. Moondream generated a coherent description of the landscape, identified two roundabouts, and detected four groups of solar panels. SamGeo3 satisfactorily segmented the Jaguaribe River channel, although it showed greater overlap in vegetation mapping. In the semantic segmentation stage, configuring the model with three classes enabled the discrimination of background, buildings, and trees. The best IoU was 0.7136, with an F1-score of 0.7490 and precision of 0.8975. The inference processed 3,465 windows and generated raster and vector outputs integrated into QGIS. GeoAI proved accessible and promising, although version control, visual validation, and larger training datasets remain necessary.

[1]) Mestrando em Recursos Hídricos: PPGECAM Universidade Federal da Paraíba, 58051-900 João Pessoa – PB, antonio.alcantara@academico.ufpb.br
[2]) Mestrando em Recursos Hídricos: PPGECAM Universidade Federal da Paraíba, 58051-900 João Pessoa – PB, felipe.daniel@academico.ufpb.br
3) Doutorando em Recursos Hídricos: PPGECAM Universidade Federal da Paraíba, 58051-900 João Pessoa – PB, abner.lins.silva@gmail.com
4) Professor Adjunto: Departamento de Engenharia Civil e Ambiental, Universidade Federal da Paraíba, 58051-900 João Pessoa – PB, davi.diniz@academico.ufpb.br

Palavras-Chave – Qgis; Pyton; GeoAi

INTRODUÇÃO

A crescente demanda por análises ambientais precisas e em larga escala impulsionou o desenvolvimento e a adoção de ferramentas avançadas de geoprocessamento, capazes de lidar com grandes volumes de dados e automatizar processos repetitivos. Entre as plataformas mais inovadoras destaca-se o Google Earth Engine (GEE), uma solução em nuvem que democratiza o acesso ao processamento geoespacial em escala planetária, permitindo análises ambientais complexas sem a necessidade de infraestrutura computacional local (Seyed et al., 2020; Tamiminia et al., 2020). O GEE oferece acesso a petabytes de imagens de satélite e integra algoritmos avançados para monitoramento ambiental, detecção de mudanças, classificação de uso do solo e resposta a desastres naturais, consolidando-se como ferramenta essencial para pesquisas ambientais globais (Velástegui-Montoya et al., 2023). Anterior a isso, o QGIS se consolidou como o sistema de informação geográfica (SIG) *open source* de análise rápida para visualização, operação e processamento, sendo leve, mais popular e robusto, oferecendo uma vasta gama de ferramentas analíticas, integração com linguagens de programação como Python e um ecossistema dinâmico de plugins que ampliam suas funcionalidades para aplicações científicas e ambientais (Rosas-Chavoya et al., 2021; SandhyaM & Prof, 2020; Szczepanek, 2012). O QGIS permite desde análises espaciais básicas até fluxos complexos envolvendo modelagem hidrológica, análise multitemporal e integração com outras plataformas *open source* e modelos de inteligência artificial.

Nos últimos anos, o avanço das técnicas de inteligência artificial (IA) revolucionou o campo do geoprocessamento ambiental. Modelos fundacionais como o Segment Anything Model (SAM), inicialmente proposto pela Meta AI, trouxeram uma nova abordagem para segmentação automática de imagens ao permitir generalização zero-shot e interação via prompts textuais ou visuais (Gui et al., 2024; Liu et al., 2024; Yan et al., 2023). Extensões do SAM adaptadas para imagens multiespectrais ou multimodais ampliam ainda mais sua aplicabilidade em cenários ambientais complexos (Liu et al., 2024). Além disso, modelos multimodais baseados em grandes modelos de linguagem (LLMs/LMMs), como EarthMarker e GeoPix, estão sendo desenvolvidos para interpretar instruções textuais junto a dados visuais geoespaciais, promovendo automação inteligente na análise ambiental (Ou et al., 2025; S. Wang et al., 2024; Zhang et al., 2024).

Nesse contexto, emerge o campo da Geospatial Artificial Intelligence (GeoAI), que integra inteligência artificial, aprendizado de máquina e aprendizagem profunda com dados e sistemas geoespaciais para ampliar a capacidade de descoberta de conhecimento geográfico (Choi, 2023). O

avanço das técnicas de IA, especialmente os modelos fundacionais como o Segment Anything Model (SAM), revolucionou o processamento e a segmentação de imagens de sensoriamento remoto, tornando possível identificar objetos e padrões em imagens aéreas ou orbitais com mínima intervenção humana. O SAM, por exemplo, utiliza uma arquitetura baseada em transformers — originalmente desenvolvida para processamento de linguagem natural — adaptada para visão computacional, sendo treinado em bilhões de máscaras e milhões de imagens. Sua capacidade de generalização permite segmentar objetos em diferentes contextos geográficos sem necessidade de treinamento adicional, operando tanto em modo automático quanto interativo via prompts como pontos, caixas delimitadoras ou descrições textuais (Gui et al., 2024; Liu et al., 2024; Osco et al., 2023). Ferramentas como o plugin SAMGeo para Python integram o SAM ao ambiente geoespacial, facilitando sua aplicação em plataformas como QGIS, onde usuários podem segmentar dados remotos e exportar resultados em formatos vetoriais ou raster sem necessidade de treinar modelos do zero (Wu & Osco, 2023)

Além do SAM, a evolução dos modelos fundacionais para GeoAI inclui arquiteturas baseadas em Vision Transformers (ViT) e abordagens multimodais que lidam com diferentes tipos e resoluções de dados geoespaciais. Modelos como SpectralGPT e Prithvi foram projetados especificamente para lidar com imagens espectrais e multiespectrais provenientes de satélites, utilizando estratégias avançadas como autoencoders mascarados (MAE) e pré-treinamento multitarefa para maximizar a eficiência na transferência para tarefas downstream — classificação, detecção e segmentação semântica (Cha et al., 2023; Hong et al., 2023; Lu et al., 2024). Um dos plugins que utilizam estas tecnologias é o GeoAI no QGIS ou pacotes Python, que assim como *SAMGeo* utilizam essas bases tecnológicas para oferecer interfaces amigáveis que permitem desde segmentação automática até integração com fluxos interativos baseados em prompts textuais ou visuais. Assim, a técnica por trás da IA do plugin GeoAI é fundamentada na combinação entre modelos fundacionais treinados massivamente (como SAM), arquiteturas transformer adaptadas ao domínio geoespacial e ferramentas que democratizam o acesso à análise inteligente de dados espaciais (Wu, 2026a; Wu & Osco, 2023).

A segmentação semântica é um dos pilares do GeoAI, permitindo a classificação pixel a pixel de imagens de sensoriamento remoto para identificar e mapear feições ambientais com alta precisão. O avanço das arquiteturas de deep learning, como U-Net, DeepLab, ResUNet-a e suas variantes, revolucionou o campo ao possibilitar a extração eficiente de características multiescalares e a integração de informações contextuais locais e globais (Diakogiannis et al., 2019; L. Wang et al., 2022). Modelos baseados em encoder-decoder, como o U-Net, tornaram-se referência para

segmentação semântica em aplicações ambientais devido à sua capacidade de reconstruir detalhes espaciais finos enquanto preservam o contexto global da cena (Jakubik et al., 2023; L. Wang et al., 2022) Além disso, funções de perda especializadas, como Dice Loss e suas variantes, têm sido empregadas para acelerar a convergência do treinamento e lidar com desafios típicos do sensoriamento remoto, como grande variação intra-classe e desbalanceamento entre classes (Diakogiannis et al., 2019). O treinamento desses modelos no contexto do GeoAI (que até a presente data deste artigo, possui *Short Paper* publicado descrevendo o *Statement of Need*) (Wu, 2026b), frequentemente recorre ao transfer learning e ao uso de modelos fundacionais pré-treinados em grandes volumes de dados não rotulados (Cui et al., 2020; Hong et al., 2023; D. Wang et al., 2024).

Deste modo, o presente trabalho objetiva é avaliar o desempenho técnico do plugin gratuito GeoAI para QGIS, analisando sua eficiência computacional e acurácia no processamento de dados geoespaciais, estando sua performance da sua metodologia por meio de um estudo de caso prático aplicado no Centro de Tecnologia, da Universidade Federal da Paraíba, e no entorno do município de João Pessoa.

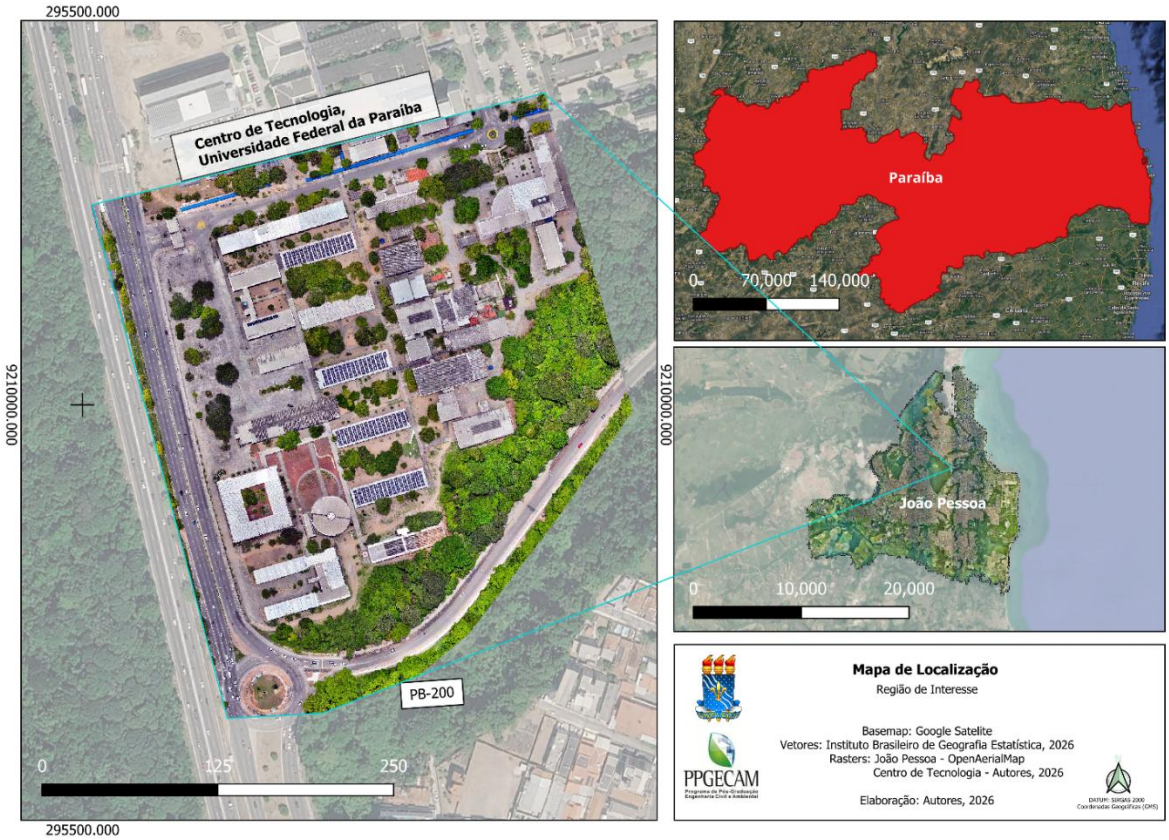
METODOLOGIA

Área de estudo

A área de estudo escolhida para a avaliação do plugin GeoAi corresponde ao Centro de Tecnologia (CT) da Universidade Federal da Paraíba (UFPB) e ao seu entorno imediato, localizado no Campus I, Cidade Universitária, bairro Castelo Branco, no município de João Pessoa, Paraíba. O Campus I situa-se aproximadamente nas coordenadas 7°06′54″ S e 34°51′47″ W (Elisei, 2017). João Pessoa possui área territorial de 210,044 km², população de 833.932 habitantes e densidade demográfica de 3.970,27 hab. km^{-2}, conforme os resultados do Censo Demográfico de 2022 (IBGE 2025). Instituído em 28 de fevereiro de 1974, o CT constitui uma unidade acadêmica voltada ao desenvolvimento de atividades de ensino, pesquisa e extensão nas áreas das Engenharias, Química Industrial e Arquitetura e Urbanismo (UFPB 2026). Para o presente estudo, o recorte espacial foi definido pela extensão do ortomosaico de alta resolução que abrange o CT e áreas adjacentes do Campus I, utilizado como base para os experimentos de interpretação visual, consultas textuais, segmentação orientada por prompts e segmentação semântica realizados no plugin GeoAI para QGIS.

A área de estudo pode ser vista no mapa de localização na Figura 1 abaixo, com destaque para o município de João Pessoa, e o Centro de Tecnologia da Universidade Federal da Paraíba.

Figura 1 - Mapa de Localização da Região de Estudo



Fonte: Autores, 2026

A metodologia deste trabalho irá se resumir em 3 principais pontos: a instalação do plugin, seguindo o manual de instalação presente na plataforma oficial do GeoAi; a descrição e correção de eventuais erros de instalação ou de performance do modelo; as configurações dos modelos, os parâmetros de treinamento, das inferências e implantações para as análises das inteligências artificiais. A documentação oficial apresenta duas formas de configuração: o instalador interno do plugin e o ambiente Pixi, sendo este último recomendado quando se deseja maior controle sobre PyTorch e CUDA. O site oficial também fornece o passo a passo para Windows, incluindo criação do projeto, configuração do pixi.toml, instalação, autenticação no Hugging Face, download do SAM 3 e abertura do QGIS pelo Pixi .

Instalação e configuração do ambiente computacional

A instalação do plugin GeoAI foi realizada em sistema operacional Windows, em equipamento dotado de GPU NVIDIA GeForce RTX 3050 Laptop, com 6 GB de memória dedicada, possibilitando a utilização de aceleração CUDA durante a execução dos modelos de inteligência artificial. O procedimento adotado seguiu a modalidade avançada de instalação por meio do gerenciador de ambientes Pixi, indicada na documentação oficial do GeoAI para situações em que se necessita de maior controle sobre as versões do Python, PyTorch, CUDA e demais dependências computacionais.

Essa alternativa foi selecionada porque a execução de modelos de aprendizagem profunda no QGIS envolve bibliotecas com requisitos específicos, cuja instalação direta no ambiente Python interno do software pode resultar em conflitos de dependências. O manual informa como requisitos o QGIS 3.28 ou superior, Python 3.10 ou superior, PyTorch e os pacotes especializados utilizados nas ferramentas do plugin, recomendando o uso do Pixi para a preparação de ambientes com suporte a GPU (*QGIS Plugin - GeoAI*, s.d.; Wu, 2026a).

Inicialmente, o Pixi foi instalado no sistema e sua disponibilidade foi verificada no PowerShell por meio do comando pixi --version, que identificou a versão 0.70.2. Em seguida, criou-se uma pasta destinada aos projetos gerenciados pelo Pixi no diretório do usuário e, dentro dela, foi inicializado o projeto denominado geo, mediante a execução dos comandos pixi init geo e cd geo. Esse procedimento criou o arquivo pixi.toml, utilizado para definir os canais de distribuição, a plataforma computacional e as versões das dependências necessárias ao ambiente. O arquivo foi configurado conforme as orientações do manual para Windows e GPU NVIDIA compatível com CUDA 12.x, contemplando Python 3.12, QGIS 3.42, PyQt 5.15, PyTorch com suporte a GPU, GeoAI, Segment-Geospatial, SAM 3 e demais bibliotecas requeridas. Posteriormente, executou-se o comando pixi install, responsável por resolver, baixar e instalar os componentes declarados no projeto, criando um ambiente isolado no subdiretório .pixi (*QGIS Plugin - GeoAI*, s.d.; Wu, 2026b).

Após a criação do ambiente, procedeu-se à verificação da compatibilidade do PyTorch com a GPU e o CUDA, conforme recomendado na documentação. Essa etapa permite confirmar se a biblioteca reconhece corretamente o dispositivo gráfico e se a aceleração por hardware está disponível. O QGIS foi então iniciado pelo comando pixi run qgis, assegurando que o programa utilizasse as versões e variáveis definidas no ambiente Pixi, em vez de uma instalação independente existente no sistema operacional. No interior do QGIS, acessou-se o menu de gerenciamento de complementos, pesquisou-se o plugin GeoAI no repositório oficial e realizou-se sua instalação e habilitação. O QGIS continuou sendo aberto pelo ambiente Pixi durante os experimentos, de modo a manter constantes as condições computacionais utilizadas nas análises (*QGIS Plugin - GeoAI*, s.d.; Wu, 2026b).

Para habilitar as ferramentas fundamentadas no Segment Anything Model 3, foi criada uma conta na plataforma Hugging Face e solicitada autorização para acessar o repositório facebook/sam3, uma vez que o modelo possui acesso controlado. Após a criação de um token pessoal, a autenticação foi realizada no terminal mediante o comando pixi run hf auth login. Depois da aprovação da solicitação de acesso, os arquivos do modelo foram obtidos por meio do comando pixi run hf download facebook/sam3. Esse fluxo — solicitação de acesso, autenticação e download —

corresponde ao procedimento indicado no manual de instalação do GeoAI para utilização do SAM 3 (*Models · Hugging Face*, s.d.; *QGIS Plugin - GeoAI*, s.d.).

Na primeira abertura de uma ferramenta do plugin, foi apresentado o instalador interno de dependências do GeoAI. Embora o QGIS já estivesse sendo executado em um ambiente Pixi, o plugin utiliza um segundo ambiente virtual, armazenado no diretório \.qgis_geoai, destinado especificamente às bibliotecas de inteligência artificial. Esse instalador detecta automaticamente a presença de GPU NVIDIA, seleciona uma distribuição do PyTorch compatível com CUDA e instala pacotes como geoai-py, segment-geospatial, SAM 3, DeepForest, OmniWaterMask e Transformers. A criação desse ambiente separado tem como objetivo isolar as bibliotecas dos modelos em relação ao Python interno do QGIS, reduzindo a possibilidade de interferência sobre os componentes nativos do sistema de informação geográfica. Após a conclusão dessa etapa e a reinicialização do programa, as ferramentas do GeoAI foram disponibilizadas para execução dos testes de interpretação multimodal, consultas textuais, segmentação orientada por prompts e segmentação supervisionada (*QGIS Plugin - GeoAI*, s.d.).

Ao término da configuração, o ambiente empregado nos experimentos foi composto pelo QGIS executado por meio do Pixi, pelo ambiente virtual interno do plugin em Python 3.12.8, pelo PyTorch 2.12.0 com suporte ao CUDA 12.6, pelo geoai-py 0.40.1 e pelo Transformers 4.56.2. O acesso ao SAM 3 permaneceu associado à conta autenticada no Hugging Face, e o QGIS foi iniciado sistematicamente pelo comando pixi run qgis, garantindo a manutenção das versões e condições computacionais utilizadas no desenvolvimento do estudo.

Problemas encontrados durante a instalação e respectivas soluções

1. Falha na vinculação do executável do QGIS durante o pixi install. Na primeira tentativa de criação do ambiente, o processo foi interrompido pela mensagem failed to link qgis-3.42.2-py312h20a3e66_0.conda. O relatório indicava que o Pixi não conseguia copiar o arquivo Library/bin/qgis.exe do cache local do Rattler para o ambiente do projeto, pois o executável esperado não estava presente no diretório do pacote. Como a resolução das dependências havia sido concluída, mas um dos arquivos necessários não havia sido corretamente extraído ou armazenado, a ocorrência foi interpretada como uma inconsistência no cache Conda ou como consequência de uma instalação parcialmente concluída. A solução consistiu na remoção do ambiente incompleto localizado na pasta .pixi, seguida da limpeza do cache Conda com o comando pixi clean cache --conda --yes. Posteriormente, o comando pixi install foi executado novamente, forçando um novo download e uma nova extração do pacote do QGIS. A segunda instalação foi concluída corretamente, confirmando que o

problema estava associado à cópia armazenada no cache. A documentação do Pixi prevê o uso da opção --conda para a limpeza específica dos pacotes Conda armazenados localmente (*pixi clean cache - Pixi*, s.d.)

2. **Negativa temporária de acesso ao repositório do SAM 3.** Após a criação da conta no Hugging Face, geração do token e autenticação pelo comando pixi run hf auth login, a tentativa de download realizada com pixi run hf download facebook/sam3 retornou a mensagem Access denied. This repository requires approval. A ocorrência não estava relacionada a erro no token ou na autenticação, mas ao fato de o SAM 3 estar disponibilizado em um repositório de acesso controlado, também denominado *gated repository*. Nessa modalidade, a autenticação identifica o usuário, porém o download depende da liberação da conta pelos responsáveis pelo modelo. Não foi necessária nenhuma alteração na instalação, no token ou nas variáveis do ambiente. A solução consistiu apenas em aguardar a aprovação da solicitação pelo próprio Hugging Face. Após a autorização, o mesmo comando foi repetido e os arquivos foram baixados normalmente. A documentação do Hugging Face estabelece que modelos controlados podem depender de aprovação automática ou manual antes que o acesso aos arquivos seja concedido (*Models · Hugging Face*, s.d.).

3. Falha na resolução das dependências internas do plugin. Na etapa de instalação das bibliotecas adicionais solicitadas pelo GeoAI, o instalador informou que as restrições de versões eram impossíveis de satisfazer. O relatório indicava que o pacote geoai-py, em versão igual ou superior a 0.39.0, dependia de uma versão recente do pacote datasets, que, por sua vez, exigia requests em versão igual ou superior a 2.32.2. Entretanto, o resolvedor encontrou apenas requests==2.28.1 no índice utilizado para a instalação dos pacotes CUDA do PyTorch e concluiu que não existia uma combinação compatível. O problema decorreu da estratégia padrão first-index do instalador uv, pela qual a resolução de determinada biblioteca é limitada ao primeiro índice em que o nome do pacote é encontrado, mesmo quando versões compatíveis estão disponíveis em outro repositório. A correção consistiu em definir temporariamente, no PowerShell, a variável $env:UV_INDEX_STRATEGY = "unsafe-best-match" antes de iniciar novamente o QGIS com pixi run qgis. Em seguida, foi acionada a opção Reinstall Dependencies. Com essa estratégia, o uv passou a considerar conjuntamente as versões disponíveis nos diferentes índices e selecionou os pacotes compatíveis, permitindo a conclusão da instalação. A documentação oficial do uv informa

que unsafe-best-match pesquisa candidatos em todos os índices configurados e seleciona a melhor versão disponível entre eles (*Package indexes | uv*, s.d.).

4. Produção de respostas incoerentes pela ferramenta Moondream. Após a instalação, os primeiros testes realizados nos modos Caption e Query produziram sequências extensas de palavras repetidas, caracteres e fragmentos sem relação com as imagens analisadas. Para verificar se o comportamento estava associado ao ortomosaico da área de estudo, o procedimento foi repetido com o raster de exemplo disponibilizado na documentação do GeoAI. Como a anomalia permaneceu, a imagem de entrada foi descartada como causa principal e as versões das bibliotecas do ambiente interno foram verificadas. A inspeção identificou PyTorch 2.12.0 com CUDA 12.6, Transformers 5.12.0, Tokenizers 0.22.2 e geoai-py 0.40.1. O problema foi associado empiricamente à incompatibilidade entre o Moondream e a versão 5.12.0 do Transformers, biblioteca responsável por componentes de carregamento, tokenização e geração textual do modelo. Para corrigir a falha, utilizou-se o executável uv do ambiente interno do plugin para instalar especificamente transformers==4.56.2. Durante a substituição, o pacote huggingface-hub foi ajustado automaticamente da versão 1.19.0 para 0.36.2. Em seguida, removeu-se o diretório de cache do modelo vikhyatk/moondream2, obrigando o sistema a realizar um novo download dos seus arquivos. Após reiniciar o QGIS por meio do ambiente Pixi, os modos Caption e Query passaram a produzir respostas coerentes com o conteúdo visual. Dessa maneira, a versão 4.56.2 do Transformers foi mantida no ambiente utilizado no restante dos experimentos.

É importante ressaltar, que as correções aqui realizadas e metodologias já descritdas ou abaixo mencionadas são válidas de replicação muito possivelmente unicamente para a versão atual instalada do plugin GeoAi (Versão 1.5.1), e para configurações de máquina iguais/semelhantes a este hardware já descritos. As falhas aqui apresentadas, bem como seus respectivos métodos de correção podem ser analisados de forma concisa na Tabela 1 abaixo:

Tabela 1 - Problemas identificados durante a instalação e procedimentos de correção

Nº	***Problema identificado***	***Causa provável***	***Procedimento de correção***	***Resultado***
1	Falha na cópia de qgis.exe durante pixi install	Pacote incompleto ou cache conda corrompido	Remoção da pasta .pixi, limpeza com pixi clean cache --conda --yes e repetição da instalação	Ambiente QGIS instalado corretamente

2	Acesso negado ao repositório facebook/sam3	Repositório controlado e solicitação ainda pendente	Aguardar a aprovação da conta pelo Hugging Face	Download liberado sem alteração do token
3	Incompatibilidade entre geoai-py, datasets e requests	Estratégia first-index do resolvedor uv	Definição temporária de UV_INDEX_STRATEGY=unsafe-best-match e reinstalação das dependências	Resolução correta dos pacotes
4	Respostas incoerentes no Moondream	Incompatibilidade com transformers 5.12.0	Instalação de transformers==4.56.2 e limpeza do cache do Moondream2	Modos *Caption*, *Query* e *Detect* normalizados

Fonte: Autores, 2026

Configuração dos modelos e procedimentos de análise

As análises de inteligência artificial foram conduzidos no plugin GeoAI integrado ao QGIS, utilizando os módulos Moondream, Segment Anything e Semantic Segmentation. Esses módulos contemplam, respectivamente, a interpretação multimodal de imagens, a segmentação orientada por prompts e o treinamento supervisionado de modelos para classificação pixel a pixel. Todas as operações foram executadas no ambiente computacional previamente configurado, com seleção automática do dispositivo de processamento, permitindo ao plugin utilizar a GPU compatível com CUDA quando disponível. Os prompts foram mantidos em língua inglesa, exatamente como inseridos na interface, com o objetivo de preservar a reprodutibilidade dos testes. O fluxo adotado seguiu as etapas e opções disponibilizadas pelo manual oficial do GeoAI para QGIS.

Inicialmente, empregou-se o modelo de visão e linguagem vikhyatk/moondream2, disponibilizado no módulo Moondream. O dispositivo de processamento foi configurado como Auto, permitindo a seleção automática entre GPU e CPU, e o ortomosaico RGB do Centro de Tecnologia da Universidade Federal da Paraíba, denominado CT, foi definido como imagem de entrada. No primeiro teste, selecionou-se o modo Caption, destinado à geração automática de uma descrição geral do conteúdo da imagem, mantendo-se o parâmetro de extensão em normal. Em seguida, o modo de análise foi alterado para Query, no qual foi inserida a pergunta “How many roundabouts?”, com a finalidade de avaliar a capacidade do modelo de interpretar a cena e contabilizar as rotatórias visualmente identificáveis no ortomosaico. Por fim, selecionou-se o modo Detect e inseriu-se o prompt “Solar Panels”, solicitando a localização espacial dos conjuntos de painéis solares por meio de caixas delimitadoras. Esses três modos correspondem às funções de descrição de imagem, resposta

visual a perguntas e detecção de objetos disponibilizadas pelo painel Moondream do GeoAI(*QGIS Plugin - GeoAI*, s.d.; Wu, 2026b).

Na segunda etapa, foi empregada a ferramenta Segment Anything com o modelo SamGeo3, correspondente à implementação geoespacial do SAM 3. Como dado de entrada, utilizou-se um recorte da região do Rio Jaguaribe, extraído de um ortomosaico de João Pessoa disponibilizado na plataforma OpenAerialMap. O backend foi configurado como meta, o dispositivo como auto e o limiar de confiança como 0,50. O modo interativo por pontos ou caixas permaneceu habilitado, embora o procedimento realizado tenha utilizado segmentação por linguagem natural. Foram mantidas as três bandas RGB padrão do raster, correspondentes às bandas 1, 2 e 3, sem aplicação de uma composição personalizada. Após o carregamento do modelo e a definição da imagem, foram executados separadamente os prompts textuais “River” e “Trees”, buscando delimitar, respectivamente, o curso d’água e a cobertura arbórea presente no recorte. A utilização de prompts textuais possibilita a segmentação de categorias descritas pelo usuário sem a necessidade de treinamento supervisionado específico para cada imagem, constituindo uma aplicação de segmentação orientada por modelos fundacionais (Wu, 2026b; Wu & Osco, 2023).

Na última etapa, foi realizada a segmentação semântica supervisionada de edificações e árvores. Para a preparação dos dados, utilizou-se o raster RGB CT_Training, correspondente a um recorte do Centro de Tecnologia, juntamente com uma camada vetorial de referência denominada ct_training. A camada continha 30 feições delimitadas manualmente, das quais oito foram classificadas como building e 22 como tree, utilizando-se o campo class para registrar a categoria de cada polígono. Os dados foram divididos em blocos de 512 × 512 pixels, com deslocamento espacial — stride — de 256 pixels, produzindo uma sobreposição de 50% entre blocos consecutivos. Essa sobreposição foi adotada para ampliar a quantidade de amostras e reduzir a perda de informações nas bordas dos recortes. O buffer foi mantido em 0,00 e o formato de exportação selecionado foi PASCAL_VOC, sendo os blocos de imagem e suas respectivas máscaras armazenados em diretórios separados, conforme o fluxo de criação de dados de treinamento do GeoAI.

O treinamento foi realizado com a arquitetura U-Net, utilizando o codificador ResNet-34 e pesos previamente treinados no conjunto ImageNet. A U-Net emprega uma estrutura encoder-decoder com conexões entre os níveis de contração e expansão, permitindo combinar informações contextuais com a recuperação dos detalhes espaciais necessários à segmentação pixel a pixel (Ronneberger et al., 2015). O ResNet-34 foi utilizado como extrator de características devido à presença de conexões residuais, que favorecem o treinamento de redes profundas (He et al., 2016), enquanto os pesos ImageNet forneceram parâmetros iniciais previamente aprendidos em um conjunto amplo de

imagens, caracterizando uma estratégia de transferência de aprendizagem (Deng et al., 2009). Foram utilizados três canais de entrada, correspondentes às bandas vermelha, verde e azul do ortomosaico, e três classes de saída: fundo, edificações e árvores. Assim, embora apenas duas categorias temáticas tenham sido delimitadas, a classe de fundo também foi contabilizada na configuração do modelo.

Os hiperparâmetros empregados foram tamanho de lote igual a 8, 50 épocas de treinamento, taxa de aprendizagem de 0,001 e separação de 20% dos pares imagem-máscara para validação. A opção de armazenamento das curvas de treinamento foi habilitada para permitir o acompanhamento posterior da perda e das métricas de validação. Não foi fornecido um checkpoint externo para ajuste fino, não se utilizou retomada de treinamento e o codificador não foi congelado, permitindo a atualização conjunta dos parâmetros do encoder e do decoder. Ao término do treinamento, utilizou-se o arquivo best_model.pth gerado pelo procedimento como modelo de entrada para a etapa de inferência.

Na inferência, foi selecionado o raster RGB CT_Input, mantendo-se a arquitetura U-Net, o codificador ResNet-34, os três canais de entrada e as três classes utilizadas durante o treinamento. O processamento foi executado por janelas de 512 × 512 pixels, com sobreposição de 256 pixels, mantendo correspondência espacial com as dimensões adotadas na criação dos dados de treinamento. O tamanho do lote foi reduzido para 4 durante a inferência, visando controlar o consumo de memória gráfica no processamento simultâneo das janelas. A opção de limiar probabilístico permaneceu desabilitada, não sendo aplicado um valor adicional de corte às previsões, e o resultado foi configurado para ser salvo em formato GeoTIFF e automaticamente adicionado ao projeto do QGIS. A geração de um mapa separado de probabilidades também permaneceu desativada. Esse procedimento completou o fluxo metodológico de criação dos dados, treinamento do modelo e aplicação da inferência sobre uma imagem geoespacial distinta, conforme a organização do painel de segmentação semântica do GeoAI (*QGIS Plugin - GeoAI*, s.d.; Wu, 2026c)

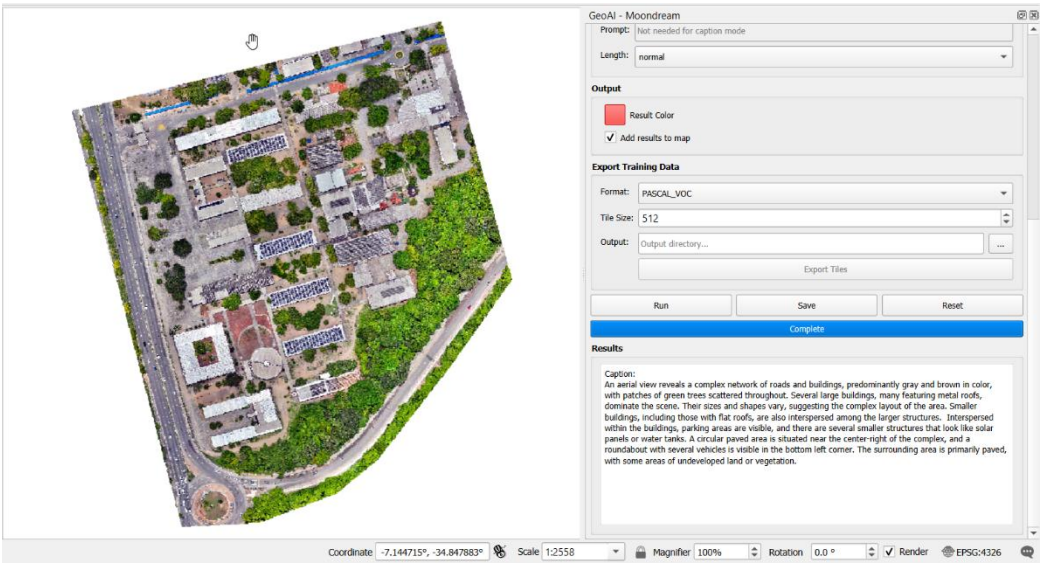
RESULTADOS E DISCUSSÕES

Análises multimodais com o Moondream

A primeira análise foi realizada no modo *Caption* do Moondream, utilizando como entrada o ortomosaico do Centro de Tecnologia da Universidade Federal da Paraíba. A descrição produzida pelo modelo reconheceu os principais componentes da cena, mencionando a presença de uma rede viária, edificações de diferentes dimensões, telhados predominantemente metálicos, árvores distribuídas entre as construções, áreas pavimentadas e espaços com vegetação. O modelo também identificou estruturas circulares e elementos semelhantes a painéis solares, demonstrando capacidade

de sintetizar, em linguagem natural, informações gerais de uma imagem aérea complexa, como pode ser visto na Figura 2 abaixo. Esse comportamento evidencia o potencial dos modelos de visão e linguagem para a interpretação preliminar de dados geoespaciais, especialmente em situações nas quais se deseja obter uma caracterização rápida do conteúdo visual sem a elaboração prévia de regras de classificação.

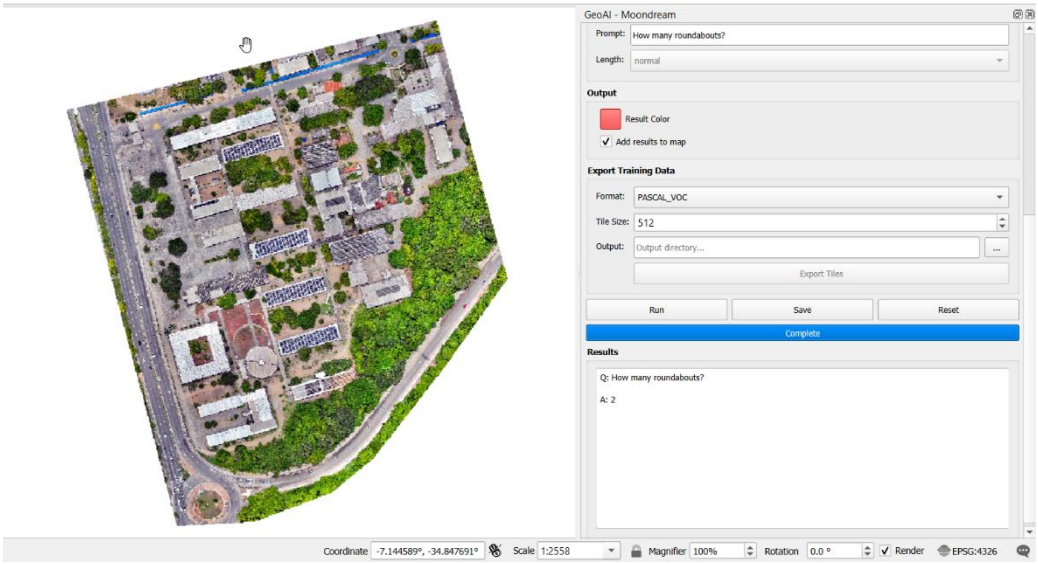
Figura 2 - Descrição 'Caption' da área de estudo



Apesar da descrição globalmente coerente, algumas expressões empregadas pelo modelo indicaram incerteza semântica. Determinadas estruturas foram descritas como elementos que poderiam corresponder a painéis solares ou reservatórios, embora os conjuntos fotovoltaicos apresentassem padrão visual relativamente característico no ortomosaico. Além disso, o texto diferenciou uma área circular pavimentada localizada no interior do campus de uma rotatória situada na parte inferior da imagem. Esse aspecto é relevante porque demonstra que o modelo foi capaz de perceber duas estruturas circulares, mas não necessariamente de atribuir a ambas a mesma função urbana. Portanto, o resultado do *Caption* foi satisfatório para a descrição geral da paisagem, embora não substitua uma interpretação especializada quando a análise depende da identificação funcional precisa dos objetos.

No teste subsequente, o modo foi alterado para *Query*, sendo apresentada a pergunta “How many roundabouts?”. O modelo respondeu que existiam duas rotatórias na imagem, como pode ser visto na Figura 3 abaixo, confirmado por inspeção visual que indica a presença de uma rotatória viária claramente definida na extremidade no centro do Ortomosaico em frente ao centro de tecnologia, enquanto a segunda estrutura circular corresponde, aparentemente, a uma área interna do campus, em frente ao Centro de Ciências Jurídicas.

Figura 3 - Resultado do Query



Ainda assim esse resultado demonstra que as respostas textuais do Moondream devem ser interpretadas como inferências probabilísticas, e não como medições cartográficas definitivas. Embora a ferramenta seja útil para consultas rápidas e exploração inicial de imagens, a contagem de objetos exige validação visual ou comparação com uma referência produzida manualmente. Em aplicações urbanas, objetos com formas semelhantes — como rotatórias, praças circulares, reservatórios e edificações arredondadas — podem ser confundidos quando o modelo utiliza principalmente atributos geométricos e contextuais da imagem.

No modo *Detect*, foi inserido o prompt “Solar Panels”, resultando na identificação de quatro objetos, representados por caixas delimitadoras sobre os principais conjuntos de painéis fotovoltaicos visíveis no ortomosaico. As quatro caixas apresentaram correspondência visual com conjuntos alongados e regularmente organizados de módulos solares, distribuídos em diferentes setores do Centro de Tecnologia. O resultado demonstrou maior consistência espacial que a consulta sobre as rotatórias, uma vez que os painéis possuem padrão de textura, tonalidade e organização repetitiva mais distinto em relação aos demais elementos da imagem. A Figura 4 6abaixo exibe a interface do plugin GeoAi apresentando a resposta, bem como as camadas do Qgis apresentando o vetor criado contendo as 4 feições das caixas de detecção dos painéis solares.

Figura 4 - Interface do GeoAi exibindo a detecção dos paineis solares



As caixas delimitadoras envolveram conjuntos completos de painéis, e não cada módulo fotovoltaico individualmente. Assim, a unidade considerada como "objeto" pelo modelo correspondeu a cada agrupamento espacial de módulos. Essa característica deve ser considerada em estudos quantitativos, pois a resposta "quatro objetos" representa quatro conjuntos ou arranjos fotovoltaicos, e não o número total de placas solares. O resultado mostra que a granularidade da detecção é determinada tanto pelo prompt quanto pela forma como o modelo organiza semanticamente os elementos da cena.

De maneira geral, os testes com o Moondream indicaram que o modelo apresentou melhor desempenho qualitativo na descrição global da imagem e na localização de objetos com padrões visuais bem definidos. Esses resultados são compatíveis com a proposta do GeoAI de integrar modelos multimodais ao processamento geoespacial, mas também reforçam que as respostas produzidas devem ser verificadas antes de sua utilização como informação temática ou quantitativa definitiva (Wu, 2026a).

Segmentação orientada por texto com o Segment Anything

Na análise com a ferramenta Segment Anything, o modelo SamGeo3 foi aplicado a um recorte do ortomosaico de João Pessoa abrangendo parte do Rio Jaguaribe e sua área urbanizada adjacente. Para o prompt "river", o modelo retornou duas regiões segmentadas, sendo a principal delas espacialmente associada ao canal do rio. A máscara acompanhou o formato alongado e sinuoso do curso d'água, indicando que o modelo conseguiu reconhecer a combinação entre geometria linear, tonalidade escura e contexto vegetado das margens.

A divisão do rio em mais de um objeto pode estar relacionada à descontinuidade visual provocada por pontes, sombras, alterações na largura do canal ou variações na resposta espectral da

água. Em imagens aéreas urbanas, a superfície de um mesmo corpo hídrico pode apresentar diferenças significativas de cor e textura devido à presença de sedimentos, vegetação, reflexos e sombreamento. Dessa forma, mesmo um elemento espacialmente contínuo pode ser interpretado pelo modelo como diferentes objetos. Ainda assim, a máscara principal apresentou correspondência satisfatória com o eixo do Rio Jaguaribe, demonstrando a capacidade do modelo fundacional de realizar segmentação sem treinamento local prévio.

Para a cobertura vegetal, o teste registrado na interface foi executado com o prompt “forest”, correspondente à intenção temática de identificar áreas arborizadas ou ocupadas por árvores. O modelo encontrou quatro objetos, produzindo máscaras sobre diferentes setores da imagem. Parte das regiões delimitadas apresentou correspondência com manchas densas de vegetação, especialmente nas proximidades do canal. Entretanto, algumas máscaras abrangeram áreas extensas e heterogêneas, indicando maior tendência à sobre segmentação do que a observada para o rio. Ambas as segmentações podem ser vistas na figura 4a e 4b abaixo, respectivamente.

Figura 5 - Segmentação do Rio e Árvores do entorno



Fonte: Autores, 2026

A maior ambiguidade do prompt relacionado à vegetação pode ser explicada pela amplitude semântica do termo *forest*. Em uma área urbana, o modelo precisa diferenciar floresta, árvores isoladas, vegetação ripária, jardins, áreas sombreadas e outros alvos com tonalidades ou texturas semelhantes. Além disso, a perspectiva vertical e a elevada heterogeneidade das imagens aéreas constituem desafios conhecidos para modelos treinados predominantemente com imagens naturais.

Estudos sobre a aplicação do SAM em sensoriamento remoto indicam que o modelo apresenta capacidade relevante de generalização, mas pode falhar em alvos com limites pouco definidos, grande variação de escala ou características específicas de imagens orbitais e aéreas (Kirillov et al., 2023). A integração implementada pelo SamGeo facilita a utilização desses modelos em dados georreferenciados, mas não elimina a necessidade de avaliar criticamente as máscaras produzidas (Wu & Osco, 2023).

Os resultados mostram, portanto, que a segmentação orientada por texto foi mais consistente para um objeto espacialmente bem definido, como o canal fluvial, do que para uma categoria ampla e internamente variável, como floresta ou cobertura arbórea. A escolha do termo utilizado no prompt influencia diretamente a interpretação do modelo. Prompts mais específicos, como “tree canopy”, “riparian vegetation” ou “dense trees”, poderiam produzir resultados diferentes e devem ser avaliados em experimentos posteriores.

Treinamento e inferência da segmentação semântica

O primeiro processamento da segmentação semântica apresentou uma inconsistência entre as métricas de validação e o resultado da inferência. Nessa execução inicial, foram exportados 3.465 blocos, dos quais apenas 548, equivalentes a 15,8%, continham feições de treinamento. Ao longo das 50 épocas, os valores de IoU e F1 permaneceram constantes em 0,8725 e 0,8800, respectivamente. Apesar desses indicadores aparentemente elevados, a inferência classificou 100% dos pixels como fundo, produzindo somente uma classe no raster final.

Esse comportamento demonstrou que os valores de validação da primeira execução não representavam efetivamente a capacidade de distinguir edificações e árvores. A configuração inicial utilizava duas classes, correspondentes a uma formulação binária entre fundo e primeiro plano, enquanto o problema proposto exigia três classes: fundo, edificações e árvores. Além disso, a baixa proporção de blocos contendo feições favorecia o predomínio do fundo no conjunto de treinamento. Em condições de forte desbalanceamento, um modelo pode alcançar métricas globais aparentemente satisfatórias ao privilegiar a classe majoritária, mas apresentar baixa capacidade de detectar classes menos representadas. Esse fenômeno é recorrente na segmentação de imagens de sensoriamento remoto, especialmente quando objetos de interesse ocupam uma parcela reduzida da cena (KAMPFFMEYER; SALBERG; JENSSEN, 2016; ZHENG et al., 2020).

Após a correção da configuração para três classes, os dados de treinamento foram novamente gerados a partir do recorte “CT_Training”. O GeoAI reconheceu as classes tree e building, reprojetou automaticamente os vetores de referência e exportou 810 blocos de 512 × 512 pixels. Desses, 549

blocos, correspondentes a 67,8% do total, continham pixels pertencentes às classes temáticas. Essa nova organização proporcionou maior representação das feições de interesse em relação à primeira execução.

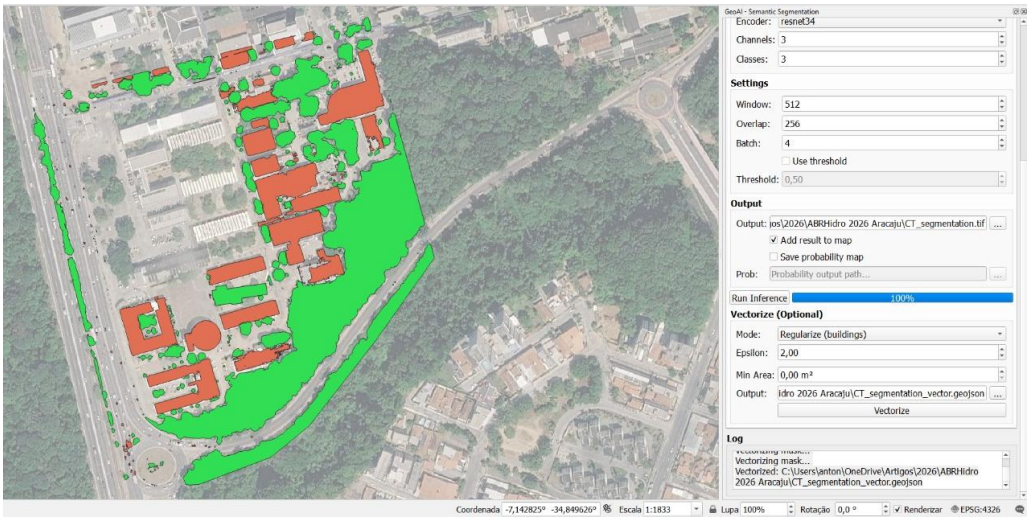
No treinamento corrigido, a função de perda diminuiu de 0,5393 na primeira época para 0,2104 na quinquagésima, indicando aprendizagem progressiva das características presentes nas imagens e máscaras. Entretanto, os valores de IoU e F1 apresentaram oscilações ao longo das épocas. O melhor IoU de validação, igual a 0,7136, foi registrado na época 48, acompanhado de F1 igual a 0,7490. Ao final do treinamento, foram obtidos IoU de 0,7018, F1 de 0,7415, precisão de 0,8975, revocação de 0,7624 e perda de validação de 0,2164.

O valor de precisão superior ao de revocação indica que, quando o modelo atribuiu um pixel às classes de interesse, houve proporção relativamente elevada de acertos; entretanto, parte dos pixels efetivamente pertencentes às classes não foi recuperada. Em termos práticos, esse comportamento é compatível com um modelo relativamente conservador, com menor tendência a produzir falsos positivos, mas ainda sujeito a omissões. O IoU próximo de 0,70 representa um resultado promissor para um experimento baseado em apenas 30 feições vetoriais originais, mas deve ser interpretado com cautela, pois as métricas apresentadas são agregadas e não permitem avaliar separadamente o desempenho para edificações e árvores.

As oscilações observadas entre épocas também podem estar relacionadas ao número limitado e desigual de amostras, composto por oito feições de edificações e 22 feições de árvores. Embora a divisão em blocos tenha aumentado o número de arquivos utilizados no treinamento, os blocos foram derivados de uma área espacial relativamente pequena e possuem sobreposição de 50%. Dessa forma, eles não constituem amostras completamente independentes. Caso blocos espacialmente vizinhos tenham sido distribuídos entre treinamento e validação, pode existir semelhança elevada entre os dois subconjuntos, aumentando a dependência espacial das métricas. A literatura ressalta que a disponibilidade limitada de referências, o desbalanceamento das classes e as diferenças entre áreas de aquisição representam desafios centrais para a segmentação semântica de imagens aéreas (TANG et al., 2023; STRAKA; GRUBER, 2024).

Na etapa de inferência, o modelo processou 3.465 janelas do raster “CT_Input” em 115,90 segundos. Foram produzidas três classes, e o fundo representou 73,0% dos pixels classificados. O raster resultante foi posteriormente convertido para vetor e submetido à regularização geométrica destinada às edificações. A Figura 6 apresenta o resultado sobreposto à imagem de referência, com as edificações representadas em vermelho e as áreas classificadas como árvores em verde.

Figura 6 - Resultado da inferência



Fonte: Autores, 2026

Visualmente, a classe de edificações apresentou boa correspondência com diversos blocos construídos, especialmente aqueles com coberturas extensas, formas retangulares e limites claramente definidos. A segmentação também identificou construções de diferentes dimensões e orientações, indicando que o modelo aprendeu padrões relacionados à geometria, textura e tonalidade das coberturas. Entretanto, algumas edificações menores ou parcialmente encobertas por árvores não foram integralmente delimitadas, o que é coerente com a revocação inferior à precisão.

A classe correspondente às árvores apresentou maior continuidade espacial e ocupou extensas regiões do setor leste e sudeste da área. Em setores com cobertura arbórea densa, o resultado acompanhou adequadamente as manchas de vegetação. Entretanto, em áreas próximas às edificações, algumas copas foram agregadas em polígonos contínuos, e regiões adjacentes com sombra ou cobertura heterogênea também foram incorporadas. Essa tendência pode estar relacionada ao maior número de feições de árvores no conjunto de treinamento, à maior área ocupada pela vegetação e à semelhança espectral entre copas, sombras e determinados tipos de superfície.

A vetorização possibilitou transformar a classificação pixel a pixel em feições editáveis no QGIS, facilitando cálculos de área e aplicações cartográficas posteriores. Contudo, o modo de regularização selecionado é especialmente apropriado para construções, cujos limites tendem a apresentar segmentos retilíneos e ângulos aproximadamente ortogonais. Para árvores e manchas vegetadas, a mesma regularização pode simplificar excessivamente limites naturalmente irregulares. Recomenda-se, portanto, que o pós-processamento seja realizado separadamente por classe, aplicando regularização ortogonal às edificações e métodos de suavização ou simplificação com preservação de forma às áreas de vegetação.

Em conjunto, os resultados demonstram que o GeoAI possibilitou executar, no ambiente do QGIS, desde interpretações multimodais e segmentações sem treinamento até a preparação de dados, o treinamento supervisionado e a inferência de um modelo U-Net. A arquitetura apresentou capacidade de extrair edificações e cobertura arbórea a partir de uma quantidade limitada de referências, confirmando a utilidade das estruturas encoder-decoder para localização espacial detalhada (RONNEBERGER; FISCHER; BROX, 2015). Entretanto, os experimentos também evidenciaram que métricas elevadas não devem ser avaliadas isoladamente, sendo necessária a inspeção visual dos resultados, o controle do desbalanceamento e a utilização de dados de validação espacialmente independentes.

Para uma avaliação quantitativa mais robusta, estudos posteriores deverão empregar amostras de validação que não participem da geração dos blocos de treinamento, além de matrizes de confusão, IoU e F1 calculados individualmente para edificações e árvores. Também é recomendável ampliar a quantidade e a diversidade espacial das feições de referência, testar funções de perda ponderadas ou baseadas no coeficiente Dice e comparar diferentes arquiteturas e codificadores. Essas medidas permitirão determinar com maior segurança a capacidade de generalização do modelo para outras áreas urbanas e diferentes condições de aquisição das imagens.

CONCLUSÃO

De modo geral, o plugin GeoAI demonstrou potencial para integrar modelos de inteligência artificial ao ambiente do QGIS, permitindo executar análises multimodais, segmentações orientadas por texto e classificação supervisionada de imagens aéreas. A ferramenta mostrou-se funcional e acessível, embora sua utilização ainda exija atenção à configuração do ambiente computacional, à compatibilidade entre bibliotecas e à validação crítica dos resultados.

1. A instalação pelo Pixi proporcionou maior controle sobre as versões do Python, PyTorch e CUDA, mas exigiu procedimentos adicionais para corrigir incompatibilidades entre pacotes e dependências.
2. Os erros encontrados durante a instalação foram solucionados sem alteração do hardware, demonstrando que parte das limitações observadas estava associada ao gerenciamento de versões e ao cache dos ambientes.
3. O Moondream apresentou desempenho satisfatório na descrição geral do ortomosaico, reconhecendo edificações, vegetação, vias, estruturas circulares e conjuntos fotovoltaicos.
4. No modo *Query*, o modelo respondeu adequadamente à consulta sobre as rotatórias, mas evidenciou que objetos visualmente semelhantes podem ser interpretados de forma ambígua.

5. O modo *Detect* identificou corretamente quatro conjuntos de painéis solares, demonstrando maior consistência para objetos com padrões geométricos, texturais e espaciais bem definidos.
6. O SamGeo3 segmentou satisfatoriamente o canal do Rio Jaguaribe sem treinamento local, confirmando o potencial dos modelos fundacionais para análises geoespaciais orientadas por texto.
7. A segmentação da vegetação apresentou maior sobreposição e generalização espacial, indicando que prompts amplos, como *forest*, podem gerar resultados menos precisos em ambientes urbanos heterogêneos.
8. A configuração inicial da segmentação semântica com duas classes resultou na classificação integral da imagem como fundo, evidenciando que a classe de fundo deve ser considerada juntamente com as classes temáticas.
9. Após a correção para três classes, o modelo U-Net com codificador ResNet-34 atingiu melhor IoU de 0,7136, F1 de 0,7490 e precisão de 0,8975, apresentando desempenho promissor diante do número reduzido de amostras.
10. A inferência permitiu distinguir edificações e cobertura arbórea, com melhor delimitação das construções regulares e maior continuidade espacial das áreas vegetadas.
11. A vetorização dos resultados ampliou a aplicabilidade do processamento, permitindo edição, cálculo de áreas e integração das classes produzidas com outras análises realizadas no QGIS.
12. Os resultados mostraram que métricas globais não devem ser interpretadas isoladamente, sendo indispensáveis a inspeção visual, a avaliação por classe e o uso de amostras independentes de validação.
13. O número limitado e desigual de feições de treinamento restringiu a capacidade de generalização do modelo, especialmente para construções menores, áreas sombreadas e limites entre árvores e edificações.
14. O GeoAI mostrou-se uma alternativa gratuita e relevante para democratizar o acesso à inteligência artificial geoespacial, reunindo diferentes modelos e etapas de processamento em uma única interface.
15. Trabalhos futuros devem ampliar e equilibrar as amostras, testar diferentes arquiteturas, funções de perda e prompts, além de avaliar o desempenho em outras áreas urbanas e condições de aquisição das imagens.

REFERÊNCIAS

Cha, K., Seo, J., & Lee, T. (2023). A Billion-scale Foundation Model for Remote Sensing Images. *ArXiv*, *abs/2304.05215*. https://doi.org/10.1109/jstars.2024.3401772

Choi, Y. (2023). GeoAI: Integration of Artificial Intelligence, Machine Learning, and Deep Learning with GIS. *Applied Sciences*. https://doi.org/10.3390/app13063895

Cui, B., Chen, X., & Lu, Y. (2020). Semantic Segmentation of Remote Sensing Images Using Transfer Learning and Deep Convolutional Neural Network With Dense Connection. *IEEE Access*, *8*, 116744–116755. https://doi.org/10.1109/access.2020.3003914

Deng, J., Dong, W., Socher, R., Li, L. J., Li, K., & Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. *2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009*, 248–255. https://doi.org/10.1109/CVPR.2009.5206848

Diakogiannis, F., Waldner, F., Caccetta, P., & Wu, C. (2019). ResUNet-a: A deep learning framework for semantic segmentation of remotely sensed data. *ArXiv*, *abs/1904.00592*. https://doi.org/10.1016/j.isprsjprs.2020.01.013

Elisei, T. (2017, fevereiro 20). *Vespas sociais (Hymenoptera, Vespidae, Polistinae) do Estado da Paraíba: Diversidade e estudo comportamental*. Universidade Federal da Paraíba. https://repositorio.ufpb.br

Gui, B., Bhardwaj, A., & Sam, L. (2024). Evaluating the Efficacy of Segment Anything Model for Delineating Agriculture and Urban Green Spaces in Multiresolution Aerial and Spaceborne Remote Sensing Images. *Remote. Sens.*, *16*, 414. https://doi.org/10.3390/rs16020414

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. https://doi.org/10.1109/CVPR.2016.90

Hong, D., Zhang, B., Li, X., Li, Y., Li, C., Yao, J., Yokoya, N., Li, H., Ghamisi, P., Jia, X., Plaza, A., Gamba, P., Benediktsson, J., & Chanussot, J. (2023). SpectralGPT: Spectral Remote Sensing Foundation Model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *46*, 5227–5244. https://doi.org/10.1109/tpami.2024.3362475

Jakubik, J., Roy, S., Phillips, C., Fraccaro, P., Godwin, D., Zadrozny, B., Szwarcman, D., Gomes, C., Nyirjesy, G., Edwards, B., Kimura, D., Simumba, N., Chu, L., Mukkavilli, S., Lambhate, D., Das, K., Bangalore, R., Oliveira, D., Muszynski, M., … Ramachandran, R. (2023). Foundation Models for Generalist Geospatial Artificial Intelligence. *ArXiv*, *abs/2310.18660*. https://doi.org/10.48550/arxiv.2310.18660

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). *Segment Anything* (arXiv:2304.02643). arXiv. https://doi.org/10.48550/arXiv.2304.02643

Liu, N., Xu, X., Su, Y., Zhang, H., & Li, H.-C. (2024). PointSAM: Pointly-Supervised Segment Anything Model for Remote Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing*, *63*, 1–15. https://doi.org/10.1109/tgrs.2025.3529031

Lu, S., Guo, J., Zimmer-Dauphinee, J., Nieusma, J., Wang, X., VanValkenburgh, P., Wernke, S., & Huo, Y. (2024). *Vision Foundation Models in Remote Sensing: A Survey*. https://consensus.app/papers/vision-foundation-models-in-remote-sensing-a-survey-lu-guo/7ad4402fb07c582d949af4d2954b7d76/

Models · Hugging Face. (s.d.). Recuperado 16 de junho de 2026, de https://huggingface.co/docs/hub/main/models

Osco, L., Wu, Q., De Lemos, E. L., Gonçalves, W., Ramos, A., Li, J., & M., J. (2023). The Segment Anything Model (SAM) for Remote Sensing Applications: From Zero to One Shot. *ArXiv*, *abs/2306.16623*. https://doi.org/10.48550/arxiv.2306.16623

Ou, R., Hu, Y., Zhang, F., Chen, J., & Liu, Y. (2025). GeoPix: A multimodal large language model for pixel-level image understanding in remote sensing. *IEEE Geoscience and Remote Sensing Magazine*, *13*, 324–337. https://doi.org/10.1109/mgrs.2025.3560293

Package indexes | uv. (s.d.). Recuperado 16 de junho de 2026, de https://docs.astral.sh/uv/concepts/indexes/

pixi clean cache—Pixi. (s.d.). Recuperado 16 de junho de 2026, de https://pixi.prefix.dev/latest/reference/cli/pixi/clean/cache/

QGIS Plugin—GeoAI. (s.d.). Recuperado 16 de junho de 2026, de https://opengeoai.org/qgis_plugin/

Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation* (arXiv:1505.04597). arXiv. https://doi.org/10.48550/arXiv.1505.04597

Rosas-Chavoya, M., Gallardo-Salazar, J., López-Serrano, P., Alcántara-Concepción, P. C., & León-Miranda, A. K. (2021). QGIS a constatly growing free and open-source geospatial software contributing to scientific development. *Cuadernos de Investigación Geográfica*. https://doi.org/10.18172/cig.5143

SandhyaM, C., & Prof, A. (2020). Exploring Opportunities with Open Source GIS. *International Journal of Engineering Research and*. https://doi.org/10.17577/ijertv9is050545

Seyed, A. G., Ahmadi, A., Kakooei, M., Moghimi, A., Mirmazloumi, S. M., Hamed, S., Moghaddam, A., Mahdavi, S., Ghahremanloo, M., Parsian, S., Wu, Q., Brisco, B., S.MohammadMirmazloumiis, & MasoudGhahremanloo. (2020). Google Earth Engine Cloud Computing Platform for Remote Sensing Big Data Applications: A Comprehensive Review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *13*, 5326–5350. https://doi.org/10.1109/jstars.2020.3021052

Szczepanek, R. (2012). *Quantum GIS – free and open source geographical information system*. *2012*, 171–182. https://doi.org/10.4467/2353737xct.14.154.2113

Tamiminia, H., Salehi, B., Mahdianpari, M., Quackenbush, L., Adeli, S., & Brisco, B. (2020). Google Earth Engine for geo-big data applications: A meta-analysis and systematic review. *Isprs Journal of Photogrammetry and Remote Sensing*, *164*, 152–170. https://doi.org/10.1016/j.isprsjprs.2020.04.001

Velástegui-Montoya, A., Montalvan-Burbano, N., Carrión-Mero, P., Rivera-Torres, H., Sadeck, L., & Adami, M. (2023). Google Earth Engine: A Global Analysis and Future Trends. *Remote. Sens.*, *15*, 3675. https://doi.org/10.3390/rs15143675

Wang, D., Zhang, J., Xu, M., Liu, L., Wang, D., Gao, E., Han, C., Guo, H., Du, B., Tao, D., & Zhang, L. (2024). MTP: Advancing Remote Sensing Foundation Model via Multitask Pretraining. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *17*, 11632–11654. https://doi.org/10.1109/jstars.2024.3408154

Wang, L., Li, R., Zhang, C., Fang, S., Duan, C., Meng, X., & Atkinson, P. M. (2022). UNetFormer: A UNet-like Transformer for Efficient Semantic Segmentation of Remote Sensing Urban Scene Imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, *190*, 196–214. https://doi.org/10.1016/j.isprsjprs.2022.06.008

Wang, S., Hu, T., Xiao, H., Li, Y., Zhang, C., Ning, H., Zhu, R., Li, Z., & Ye, X. (2024). GPT, large language models (LLMs) and generative artificial intelligence (GAI) models in geospatial science: A systematic review. *International Journal of Digital Earth*, *17*. https://doi.org/10.1080/17538947.2024.2353122

Wu, Q. (2026a). GeoAI: A Python package for integrating artificial intelligence with geospatial data analysis and visualization. *J. Open Source Softw.*, *11*, 9605. https://doi.org/10.21105/joss.09605

Wu, Q. (2026b). GeoAI: A Python package for integrating artificial intelligence with geospatial data analysis and visualization. *Journal of Open Source Software*, *11*(118), 9605. https://doi.org/10.21105/joss.09605

Wu, Q. (2026c). GeoAI: A Python package for integrating artificial intelligence with geospatial data analysis and visualization. *Journal of Open Source Software*, *11*(118), 9605. https://doi.org/10.21105/joss.09605

Wu, Q., & Osco, L. (2023). samgeo: A Python package for segmenting geospatial data with the Segment Anything Model (SAM). *J. Open Source Softw.*, *8*, 5663. https://doi.org/10.21105/joss.05663

Yan, Z., Li, J., Li, X., Zhou, R., Zhang, W., Feng, Y., Diao, W., Fu, K., & Sun, X. (2023). RingMo-SAM: A Foundation Model for Segment Anything in Multimodal Remote-Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing*, *61*, 1–16. https://doi.org/10.1109/tgrs.2023.3332219

Zhang, W., Cai, M., Zhang, T., Yin, Z., Li, J., & Mao, X. (2024). EarthMarker: A Visual Prompting Multimodal Large Language Model for Remote Sensing. *IEEE Transactions on Geoscience and Remote Sensing*, *63*, 1–19. https://doi.org/10.1109/tgrs.2024.3523505