UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE SISTEMAS DE INFORMAÇÃO – BACHARELADO

UTILIZAÇÃO DE IMAGENS DE SATÉLITE PARA DETECÇÃO DE DESLIZAMENTOS DE TERRA ATRAVÉS DE TÉCNICAS DE VISÃO COMPUTACIONAL

HENRIQUE HARUDA GOLLNICK

BLUMENAU
2023

HENRIQUE HARUDA GOLLNICK

UTILIZAÇÃO DE IMAGENS DE SATÉLITE PARA DETECÇÃO DE DESLIZAMENTOS DE TERRA ATRAVÉS DE TÉCNICAS DE VISÃO COMPUTACIONAL

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Sistemas de Informação do Centro de Ciências Exatas e Naturais da Universidade Regional de Blumenau como requisito parcial para a obtenção do grau de Bacharel em Sistemas de Informação.

Prof. Aurélio Faustino Hoppe - Orientador

BLUMENAU
2023

Esta página deverá ser substituída pela folha de assinaturas entregue na Banca.

Digitalize a folha e cole aqui para a entrega da versão final do TCC.

Atenção: não ultrapasse as margens!

Dedico este trabalho à minha família e a todos os meus colegas e amigos que me apoiaram e auxiliaram em todo o processo de graduação, que se não fosse pelo apoio e incentivo deles, eu não teria trilhado este caminho do conhecimento.

AGRADECIMENTOS

Agradeço primeiramente a minha família e aos meus colegas e alunos, que me deu todo o suporte e motivação necessária para a conclusão deste trabalho e do curso, sem eles este feito não seria possível.

A empresa T-Systems que me proporcionou toda a flexibilidade e compreensão em momentos de tensão e nervosismo nesta etapa tão importante da minha carreira e vida profissional.

Ao professor Aurélio, que me acompanhou nesta caminhada final me orientando e proporcionando os conhecimentos necessários para a conclusão deste trabalho com excelência.

RESUMO

Desastres caracterizam-se como uma grave interrupção do funcionamento de uma comunidade ou sociedade envolvendo perdas e impactos humanos, materiais, econômicos ou ambientais generalizados, que excede a capacidade da comunidade ou sociedade afetada de lidar com seus próprios recursos. Com foco no município de Blumenau, este trabalho detalha o processo de detecção de deslizamentos de terra utilizando técnicas de visão computacional a partir de imagens providas por sensoriamento remoto. Foram utilizadas 100 imagens do CPRM coletas pelo software ArcGIS e 475 imagens coletadas pelo Google Earth Pro para a criação do dataset. Posteriormente, aplicou-se técnicas de *data augmentation* e *preprocessing* para aumentar sua abrangência, totalizando 954 imagens utilizadas para treino e validação. Para a detecção de áreas com deslizamentos utilizou-se modelo YOLOv8 com a reutilização de um modelo pré-treinado do *dataset* Microsoft COCO. O modelo foi desenvolvido com a linguagem de programação Python na versão 3.7 e com a biblioteca Keras e PyTorch. Nas imagens foram definidas duas classes de objetos: *landslide* e *fissure*. Como resultado, obteve-se uma precisão de até 95,9% e de 78,7% de recall. Contudo, comprova-se que o modelo pode ser uma alternativa viável quanto a detecção de áreas aos quais ocorreram deslizamentos de terra.

Key-words: Deslizamentos. Desastres. Detecção. Visão computacional. YOLOv8.

ABSTRACT

Disasters are characterized as a severe disruption to the functioning of a community or society, involving widespread human, material, economic, or environmental losses and impacts that exceed the affected community or society's ability to cope with their own resources. Focusing on the municipality of Blumenau, this study details the process of landslide detection using computer vision techniques from remotely sensed images. One hundred images from CPRM collected through ArcGIS software and 475 images collected from Google Earth Pro were used to create the dataset. Subsequently, data augmentation and preprocessing techniques were applied to increase its coverage, resulting in a total of 954 images used for training and validation. For landslide detection, the YOLOv8 model was utilized, reusing a pre-trained model from the Microsoft COCO dataset. The model was developed using Python programming language version 3.7 and the Keras library and PyTorch. Two classes of objects were defined in the images: "landslide" and "fissure". As a result, a precision of up to 95.9% and a recall of 78.7% were achieved. However, it is proven that the model can be a viable alternative for the detection of areas affected by landslides.

Keywords: Landslides. Disasters. Detection. Computer vision. YOLOv8.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

CNN – Convolutional neural networks

FN – False Negative

FP – False Positive

G-BNECK – Ghost bottleneck

GCONV – Group convolution

OBIA – Object-based-image

R – Recall

RF – Requisito Funcional

RNF – Requisito Não Funcional

P – Precision

TN – True Negative

TP – True Positive

CPRM – Repositório Institucional de Geociências

UNDRR – United Office for Disasters Risk Reduction

SUMÁRIO

1 INTRODUÇÃO

Atualmente, os desastres naturais constituem umas das principais ameaças. Afinal, segundo o United Office for Disasters Risk Reduction (UNDRR), o risco está aumentado numa taxa sem precedentes. O agravamento dos impactos dos desastres naturais se manifesta tanto de forma extensiva pelo aumento da escala de abrangência, quanto pela intensidade de destruição. Este processo pode ser atribuído a um conjunto de fatores, incluindo, por um lado, o aumento da frequência e intensidade de eventos; mas também, pela exposição das populações aos impactos. Neste sentido, considerando em termos globais é possível identificar quatro tendências:

a) a ocorrência de desastres vem aumentando anualmente;
b) o número de pessoas afetadas por desastres está crescendo;
c) as perdas estão se tornando mais custosas;
d) as regiões mais pobres são as mais impactadas.

Isto significa que a gestão dos desastres vem se tornando cada vez mais necessária, porém sempre menos suficiente. Em termos analíticos, pode-se diferenciar dois tipos de abordagens: (i) **foco no evento**: a preocupação se concentra no estudo das características físicas deflagradoras dos desastres (TOBIN; MONTZ, 1997); (ii) **foco no impacto**: a atenção se concentra na investigação do padrão de organização das respostas da comunidade (TIERNEY, 2020). Neste sentido, estes estudos permitiram estabelecer o princípio de continuidade entre o Tempo-1 (Pré-impacto) e o Tempo-2 (Pós-impacto). A hipótese subjacente indica que as condições de vulnerabilidade existentes no Tempo-1 se convertem em destruição no Tempo-2. Ou seja, a destruição causada por um desastre reflete dois fatores: (i) caracterização inadequada do fenômeno; (ii) incapacidade de implementar medidas consistentes (MATTEDI, 2017). Atualmente, a atenção se voltou para os dois fenômenos que se encontram relacionados.

Por um lado, este processo está relacionado ao padrão predominante de ocupação do espaço e utilização dos recursos. Neste sentido, a urbanização descontrolada, a expansão das atividades agrícolas e pecuárias provocam uma dupla concentração da população: nas maiores cidades e nas áreas de risco. O aumento da população em áreas urbanas aumenta a demanda por recursos naturais e infraestrutura. Entre os principais efeitos deste processo destacam-se a crescente impermeabilização do solo, a redução de áreas verdes, a mudança do padrão de vazão dos rios, a ocupação de áreas de encostas, entre outros fatores. Além disso, a construção de infraestrutura e a diminuição de áreas naturais contribuem decisivamente para o agravamento dos desastres naturais. Ou seja, à medida que as áreas urbanas crescem, elas podem expandir as

áreas de risco, aumentando a exposição a desastres naturais como inundações, deslizamentos de terra, furacões e terremotos.

Da mesma forma, destacam-se as mudanças climáticas. As mudanças climáticas recentes vêm alterando o padrão de ocorrência dos desastres naturais (IPCC, 2012). O aumento da temperatura a intensifica secas, incêndios florestais, ondas de calor, mas também enchentes, deslizamentos de terra, ciclones tropicais. Neste sentido, a mudança no estado do clima acaba alterando a frequência, intensidade, extensão e duração eventos climáticos. As variações são provocadas tanto por processos internos naturais, quanto por processos externos antropogênicos. Com o desenvolvimento da ciência do clima nos últimos anos descobriu-se que a temperatura e a precipitação são preditores muito importantes dos impactos provocados por desastres. O efeito combinado destes fenômenos terá profundos impactos na saúde, alimentação, segurança, infraestrutura e economia de diversas regiões (MORA *et al.* 2018). Portanto, as mudanças climáticas tendem a tornar os desastres naturais mais frequentes e intensos.

Por isto, paradoxalmente as medidas de proteção são cada vez mais necessárias, porém sempre insuficientes. Ou seja, apesar dos esforços de ações voltadas a prevenção, preparação, resposta e recuperação existentes não são suficientes para mitigar os impactos dos desastres. Isto acontece porque a gestão dos desastres deve ser realizada de forma integrada e coordenada, favorecendo a participação da população. Entre os aspectos mais sensíveis deste processo destacam-se, principalmente, falhas de comunicação, baixa coordenação, falta de supervisão e, sobretudo, falta de flexibilidade. Isto tem levado ao desenvolvimento e implementação de políticas desatualizadas que não atendem aos desafios atuais. E, é por isto também que apesar do aumento de investimento de medidas estruturais de proteção e políticas de gestão dos desastres verifica-se o agravamento dos impactos (COPPOLA, 2011). Ou seja, as políticas de gestão dos desastres naturais exigem recursos financeiros, materiais, informacionais e humanos para serem implementadas adequadamente.

No Vale do Itajaí, os desastres naturais acompanham o processo de formação e de desenvolvimento da região (MATTEDI, 1999). Os primeiros registros remontam ao início de ocupação em 1852. Ao longo deste período registram-se 77 ocorrências, considerando apenas a cidade de Blumenau como referência (MATTEDI, 1999). Ao mesmo tempo, a região tem sido objeto também da implantação de um arcabouço verdadeiramente amplo de medidas de confrontação. Porém, quando se analisam os dados disponíveis, verifica-se uma tendência de agravamento do problema nas últimas décadas. Além disso, com o tempo o problema dos desastres que estava restrito a Blumenau acabou se generalizando para toda a área da bacia

hidrográfica. Neste sentido, pode-se dizer que a região vem passando por um crescente processo de vulnerabilização aos impactos dos desastres naturais.

Fernandes (2020) define que o termo “desastre socioambiental” se formula a partir de desastre naturais em conjunto com características tais como: vulnerabilidade da comunidade, ocupação, estrutura que baseia estas moradias e gerenciamento de risco da gestão local. Em Blumenau, segundo Holetz (2007), muitas famílias optam por construírem suas residências em áreas de risco, principalmente devido ao fator econômico que as impedem de adquirirem terrenos em áreas dentro dos padrões técnicos de segurança. O autor ainda complementa que os terrenos propícios à ocupação possuem valores elevados, impossibilitando o acesso da população com baixa renda que se desloca para as áreas de risco.

Segundo consta no guia de prevenção de risco de deslizamento em encostas (CARVALHO; GALVÃO, 2006, p. 37-38), uma das ações de fiscalização e controle de risco que possuem um dos melhores resultados é a realização de vistorias periódicas e sistemáticas em todas as áreas de risco por equipes técnicas. Estas servem para observar a evolução de situações de risco, identificar processos destrutivos, e orientar moradores sobre ações preventivas e obras corretivas. Contudo, este processo demanda grande quantidade de mão de obra e recursos, principalmente para atender o crescimento populacional de Blumenau. Neste sentido, a Defesa Civil de Blumenau possui a infraestrutura necessária para fazer o processo de análise, contudo esta não é a realidade de outros municípios com menor quantidade de recursos disponíveis. Ou seja, muitas áreas de risco da região não são monitoradas. Além disso, a Defesa Civil de Blumenau utiliza de imagens aéreas e de satélite para detectar fissuras em áreas de risco. Entretanto, o processo requer a análise manual das imagens capturadas por um especialista, demandando um tempo considerável para analisar o contexto da cidade inteira. Por outro lado, atualmente, pode-se encontrar alguns trabalhos que buscam realizar a identificação de deslizamentos de terra utilizando tecnologias como o *deep learning* e visão computacional através de fotografias e imagens de satélite.

Neste sentido, se levanta as seguintes questões de pesquisa: (i) é possível detectar deslizamentos e fissuras no solo a partir de imagens de satélite e de técnicas de aprendizado de máquina? (ii) os dados obtidos favorecem a caracterização de possíveis regiões suscetíveis a deslizamentos de terra? (iii) é possível realizar detecção de deslizamentos de terra com imagens de alta resolução em que seria possível não somente a detecção do deslizamento, como também a análise de seus derredores? (iv) é possível realizar o treino de um modelo de visão computacional para a detecção de deslizamentos com um número pequeno de imagens presentes no *dataset*?

1.1 OBJETIVOS

O objetivo deste trabalho é disponibilizar um protótipo que seja capaz de realizar a demarcação de fissuras no solo, as quais podem resultar em possíveis deslizamentos de terra.

Os objetivos específicos são:

a) analisar a possibilidade da detecção de fissuras no solo a partir de imagens e de técnicas de aprendizado de máquina;
b) analisar se os dados obtidos favorecem a caracterização de possíveis áreas suscetíveis a deslizamentos de terra;
c) analisar a possibilidade de realizar detecção de deslizamentos de terra com imagens de alta resolução em que seria possível não somente a detecção do deslizamento, como também a análise de seus derredores;
d) analisar a possibilidade de realizar o treino de um modelo de visão computacional para a detecção de deslizamentos com um número pequeno de imagens presentes no dataset;
e) validar e analisar o tempo de resposta da segmentação e detecção de fissuras e deslizamentos, assim como sua assertividade em relação ao processo manual.

1.2 ESTRUTURA

A estrutura do trabalho está divido em quatro capítulos. No primeiro capítulo está descrita a introdução do trabalho juntamente com os objetivos e objetivos específicos. No segundo capítulo está detalhada a fundamentação teórica, contendo os principais tópicos do trabalho e os trabalhos correlatos encontrados. O terceiro capítulo descreve o desenvolvimento da ferramenta, contendo as especificações funcionais e não funcionais da aplicação, técnicas e bibliotecas utilizadas em cada etapa, implementação do protótipo e apresentação dos resultados encontrados. Por fim, o quarto e último capítulo descreve a conclusão do trabalho, alinhando as expectativas e resultados, juntamente com as limitações e sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo descreve brevemente os assuntos que fundamentarão o estudo a ser apresentado. A seção 2.1 aborda movimento de massa. Por fim, a seção 2.2 detalha o funcionamento do modelo de visão computacional YOLOv8.

2.1 MOVIMENTO DE MASSA

O termo “movimento de massa” “deslizamento de terra” descreve uma ampla variedade de processos que resultam no movimento descendente e externo de materiais formadores de encostas, incluindo rocha, solo, preenchimento artificial ou uma combinação deles (UNITED STATES GEOLOGICAL SURVEY, 2004).

De acordo com Jongmans *et al.* (2009), define-se movimentos de massa como sendo uma área geográfica que pode ser propensa a, ou ter experimentado, o movimento de massa de material geológico em declive. Segundo Bessa (2015), movimentos de massa são caracterizados como movimentos do solo, rochas, e/ou vegetações ao longo da vertente sob a qual se aplica a ação direta da gravidade. Além disso, os deslizamentos estão diretamente correlacionados com a evolução geomorfológica, em especial nas regiões serranas ou de grande atividade tectônica. A autora ainda ressalta que embora que os movimentos de massa sejam naturais, o crescimento urbano tem sido desfavorável quanto ao planejamento adequado do solo. Bigarella (2007) descreve que estes movimentos de massa normalmente se relacionam com a presença de água ou gelo, diminuindo a resistência da rocha ou do solo de modo a aumentar a plasticidade e induzir um estado de fluidez no solo.

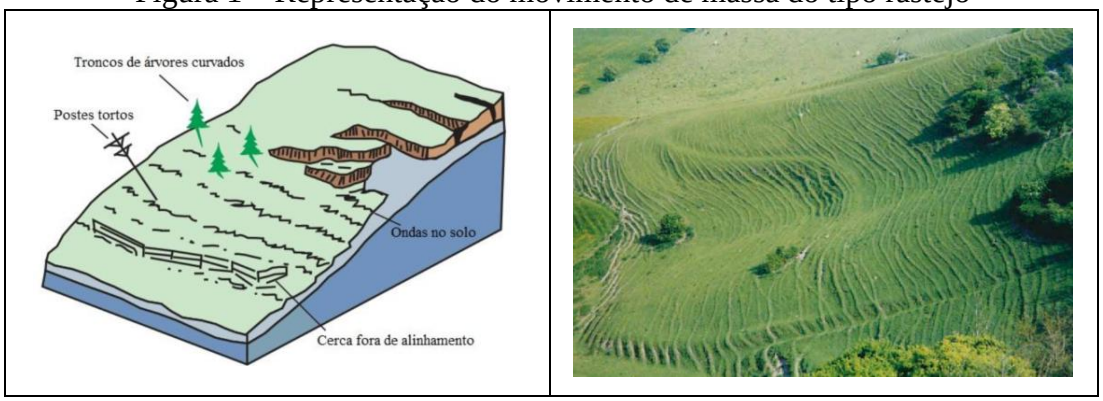
Para Augusto Filho (1992) existem várias maneiras de classificar movimentos de massa, em sua maioria, essas classificações se baseiam em quatro categorias principais: velocidade, planos de deslocamento, volume de material, geometria e o material envolvido em questão. Desta maneira, o autor classifica estes movimentos em quatro classes de processos, sendo elas: rastejos (*creep*), escorregamentos (*slides*), quedas (*falls*) e corridas (*flows*).

De acordo com Brasil (2007), rastejos (*creep*) são movimentos lentos, os quais trabalham quantidades de massa de materiais maiores, resultando em um deslocamento menor, movimentando apenas milímetros ou centímetros ao ano. Highland e Bobrowsky (2008) afirmam que rastejos são os tipos mais comuns de deslizamentos, de modo a frequentemente preceder outros tipos mais rápidos e danosos. Para Bessa (2015), os rastejos se classificam em (i) **sazonal**, ocorrendo no interior ou no fundo do solo devido a alterações sazonais, tal como por exemplo a alternância da temperatura; (ii) **contínuo**, onda a tensão de cisalhamento

sobrepassa a resistência do material em questão; e (iii) **progressivo**, onde os taludes apresentam um ponto de ruptura, de modo a gerar outros tipos de movimento do terreno.

Highland e Bobrowsky (2008) afirmam que a causa do rastejo pode possuir motivos como por exemplo: condição climáticas, químicas ou físicas, vazamento de tubulação, drenagem ineficiente, tipos de construções desestabilizadoras etc. Deste modo, Tominaga (2012) ressalta que o maior motivo para o movimento de massa em rastejos é a ação da gravidade, correlacionada com eventos climáticos, especialmente variações na temperatura e humidade. Highland e Bobrowsky (2008) afirmam que alguns indicadores de possíveis rastejos são indicados por curvas em troncos de arvores, cercas e postes, ondas ou cristas na superfície do solo, conforme pode ser visto na Figura 1.

Figura 1 – Representação do movimento de massa do tipo rastejo



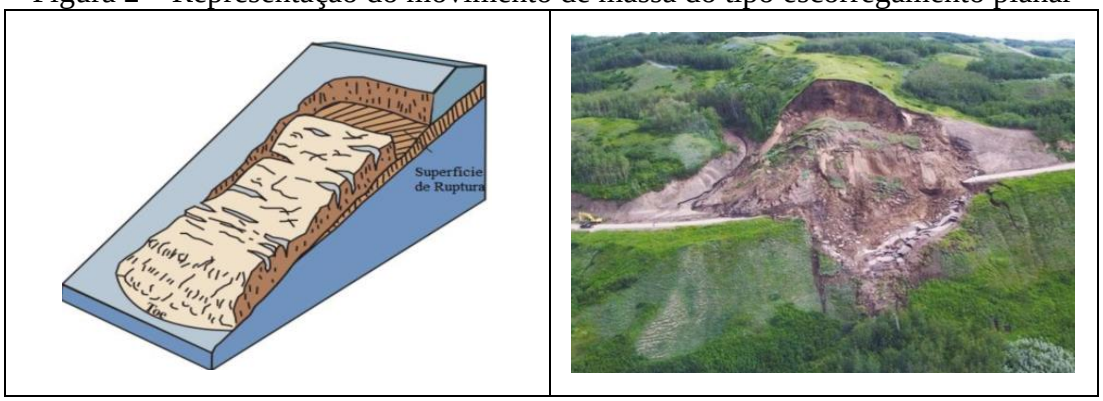
Fonte: Bessa (2015).

De acordo com Tominaga (2012) escorregamentos (*slides*) denominam movimentos rápidos de porções de terreno. Bessa (2015) afirma que a velocidade da movimentação de massa dependerá da inclinação da superfície onde está ocorrendo o escorregamento, da causa pela qual este veio a ocorrer e da natureza do terreno. Deste modo, terrenos homogêneos tendem a apresentar movimentos mais bruscos. Brasil (2007) classifica os escorregamentos/deslizamentoss em três categorias: (i) **planares ou translacionais**; (ii) **circulares ou rotacionais**; e (iii) **em cunha**. Highland e Bobrowsky (2008) complementam que fatores como: encharcamento do solo pelas chuvas, degelo, inundações, aumento do nível de água devido a irrigações, vazamentos de tubulações ou distúrbios relacionados à ação do homem, erosão regressiva e terremotos ou tremores são algumas das principais causas de escorregamentos.

Brasil (2015) afirma que os escorregamentos do tipo planar ou translacional são os mais comuns dentre todos os tipos de movimento de massa. Estes ocorrem gerando rupturas planares devido a heterogeneidade das superfícies e dos solos e rochas, apresentando descontinuidades mecânicas e/ou hidrológicas derivadas de processos geológicos, geomorfológicos ou

pedológicos. Ainda de acordo com Brasil (2007), os principais materiais transportados por escorregamentos translacionais são constituídos por rochas e por partes do solo. Fernandes e Amaral (1996) afirmam que morfologicamente, escorregamentos planares possuem a característica de serem rasos, com plano de ruptura, e em sua grande maioria entre 0,5 metros de profundidade e 5 metros de profundidade, possuindo extensões mais longas em seu comprimento. De acordo com Highland e Bobrowsky (2008), esses movimentos de massa podem acabar progredindo distâncias consideráveis, caso a superfície em questão acabe por ter um índice de inclinação maior. Os autores ainda indicam que a velocidade média de um escorregamento pode variar entre 1,5 metros ao mês e 1,5 metros ao dia. A Figura 2 exemplifica o escorregamento planar.

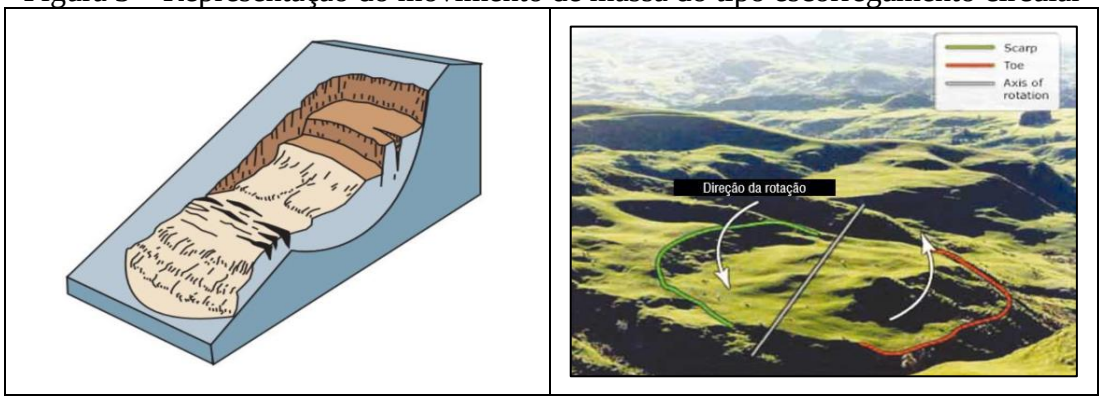
Figura 2 – Representação do movimento de massa do tipo escorregamento planar



Fonte: Bessa (2015).

Fernandes e Amaral (1996) citam que escorregamentos circulares ou rotacionais caracterizam-se por rupturas curvas na superfície onde está acontecendo o movimento no solo. Os autores ainda indicam que este tipo de movimento normalmente está correlacionado com a existência de solos espessos e homogêneos, tendo como exemplo as alterações presentes em rochas argilosas. Uma das principais causas para a ocorrência deste tipo de movimentação de massa é a existência de cortes nas bases destes, cita-se a construção de estradas ou edifícios, tal como a erosão fluvial. Brasil (2008) cita que este tipo de deslizamento tende a possuir um raio menor que os apresentados pelos deslizamentos translacionais. Deste modo, Guindicini e Nieble (1976) afirmam que estes tipos de deslizamento são frequentes em territórios no sul e sudeste brasileiro. Estes podem ter consequências catastróficas, devido ao fato de deslizarem o solo residual o qual recobre as rochas ao longo de uma superfície. A Figura 3 apresenta o escorregamento circular.

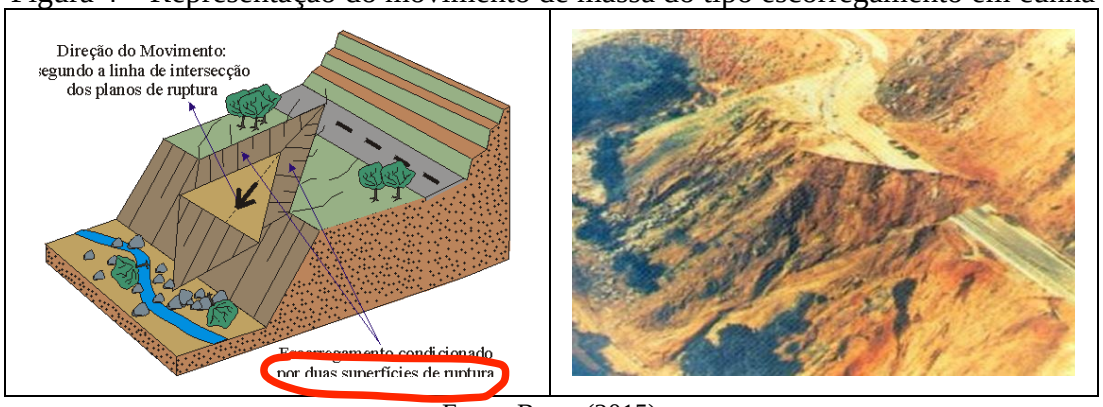
Figura 3 – Representação do movimento de massa do tipo escorregamento circular



Fonte: Bessa (2015).

De acordo com Brasil (2007), deslizamentos em cunha são diretamente associados a saprólitos e maciços rochosos, apresentando fraqueza desfavorável na estabilidade ao longo do eixo de intersecção do plano. Godoy (2005) afirma que o eixo de movimentação ocorre na intersecção destes planos de fraqueza. Portanto, as causas destes são caraterizadas por poro-pressões na superfície de escorregamento, como também pela pressão hidrostática em fendas, fissuras ou descontinuidades de maneira geral, conforme exemplifica a Figura 4. Brasil (2007) ainda cita que estes ocorrem comumente em taludes e encostas.

Figura 4 – Representação do movimento de massa do tipo escorregamento em cunha



Fonte: Bessa (2015).

De acordo com Guindicini e Nieble (1976), corridas (*flows*) são formas de rápido escoamento de massa e que são normalmente hidrodinâmicos, causada pela perda do atrito interno devido a destruição da estrutura causada pela presença de água em seu material. Brasil (2007) cita que são ocorrem normalmente a partir de escorregamentos em encostas, e tendem a mover volumes grandes de material, escoando este por um longo percurso. Guindicini e Nieble (1976) afirmam que a massa de solo e rocha podem acabar fluindo como líquido, caso atinjam certo nível de fluidez, devido a adição de água ou de materiais arenosos ou argilosos. Highland e Bobrowsky (2008) descrevem que estes fenômenos são raros quando comparados com outras tipos de movimentação de massa, contudo tendem a provocar consequências de magnitudes

superiores devido ao seu extenso raio de alcance e poder destrutivo. Consequentemente, estes movimentos de massa alcançam altas velocidades e incorporam elementos diversos pelo caminho tais como por exemplo matacões e outros tipos de fragmentos, conforme ilustra a Figura 5.

Figura 5 – Representação do movimento de massa do tipo corrida

Fonte: Bessa (2015).

De acordo com Bigarella (2007), certas condições auxiliam ocorrências de movimentação de massa, cita-se: estruturas geológicas (litologia, padrões de fraturas e diaclases, coesão, manto de intemperismo), declividade da vertente (forma topográfica), regime das chuvas, vegetação e atividades antrópicas. Segundo Bessa (2015), o regime de chuvas é outro causador das movimentações de massa, onde índices pluviométricos elevados tendem a provocar saturação do solo e das rochas, reduzindo a resistência e então causando deslizamentos.

Segundo Ullo *et al.* (2020), Yang *et al.* (2022) e Fu *et al.* (2022), no Brasil, as áreas de risco de movimento de massa são monitoradas por órgãos como a Defesa Civil, através de imagens de satélite, fotografias, e imagens aéreas as quais são manualmente captadas e analisadas por profissionais. Ainda de acordo com os autores, a ocorrência de movimento de massa é evitável, incluindo seus danos econômicos e humanos, por meio da implementação de políticas de planejamento urbano e territorial. Portanto, a promoção de políticas públicas em setores como habitação e assistência social possibilitam a atenuação dos estragos causados pelos deslizamentos de terra, além de resultar em melhoria na qualidade de vida da população.

2.2 DETECÇÃO DE OBJETIVOS ATRAVÉS DO YOLO

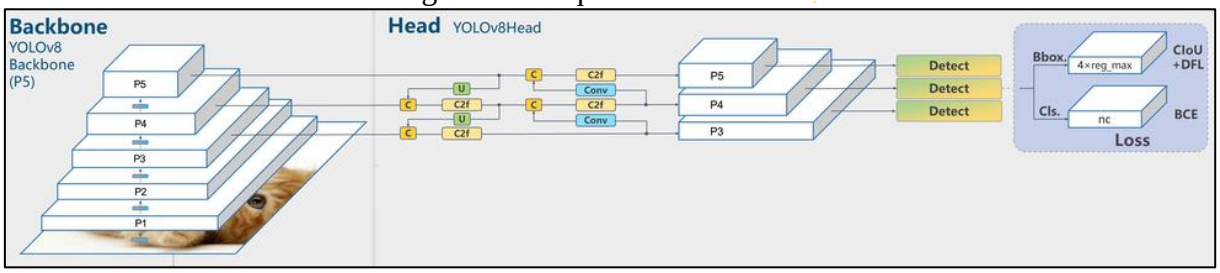
Em 2018, Redmon *et al*. (2018) propôs a arquitetura de rede neural You Only Look Once (YOLO) para reconhecimento e classificação de objetos em imagens. O foco desta arquitetura é a velocidade de detecção, sendo capaz de fornecer reconhecimento em tempo real.

De acordo com Gündüz, Mehmet e Gültekin (2023), o modelo YOLO é um tipo de detector de estágio único baseado em uma rede neural convolucional (CNN). Este é um algoritmo de detecção de objetos em tempo real que usa uma única rede neural convolucional para dividir uma imagem em *grids*, com cada *grid* fazendo previsões sobre *bouding-boxes* e pontuações de confiabilidade.

Segundo Lin *et al*. (2002), o YOLO evoluiu ao longo do tempo, com diferentes versões, como YOLOv3, YOLOv4 e YOLOv5, YOLOv7 e a mais recente YOLOv8, a qual foi lançada em janeiro de 2023. Vale ressaltar que o modelo apresentou grandes avanços em suas versões. YOLOv3 por exemplo é uma versão melhorada do modelo YOLO que pode realizar localização e classificação em tempo real apenas com a ajuda de uma rede neural (KIVRAK e GÜRBÜZ, 2022). O YOLOv8 é um novo modelo de visão computacional de última geração construído pela Ultralytics, os criadores do YOLOv5. Deste modo o modelo YOLOv8 contém suporte pronto para uso para tarefas de detecção, classificação e segmentação de objetos (ULTRALYTICS, 2023).

A arquitetura do YOLOv8 se baseia em versões previas dos modelos de detecção de objetos YOLO. Ele utiliza uma rede neural totalmente convolucional que pode ser dividida em duas partes principais: o *backbone* e o *head*. O *backbone* do YOLOv8 é uma versão modificada da arquitetura CSPDarknet53, introduzida na versão YOLOv4 (BOCHKOVSKIY *et al.* 2020). Essa arquitetura consiste em 53 camadas convolucionais e emprega uma técnica chamada de conexões parciais entre estágios para melhorar o fluxo de informações entre as diferentes camadas da rede. A Figura 6 demonstra a arquitetura do YOLOv8 desenvolvida por Bochkovskiy *et al.* (2020).

Figura 6 – Arquitetura YOLOv8.



Fonte: adaptado de Ultralytics (2023).

A arquitetura YOLO inicialmente recebe uma imagem de entrada, esta então é processada através de uma rede neural convolucional chamada de *backbone*. Em casos de classificação de imagens, a saída final da rede é suficiente para fazer uma previsão. Em contrapartida, na detecção de objetos não somente é necessário identificar a classe do objeto, como também a área exata na imagem que ele ocupa, denominada *bouding-box*

(BOCHKOVSKIY *et al.* 2020). Em síntese, é necessário sortear cuidadosamente as camadas de recursos do *backbone,* isto acontece na parte do algoritmo denominada *neck.* Além disso, os algoritmos de detecção de objetos podem ser categorizados em dois tipos: *one-stage detectors* e *one-two-stage detectors*. *Two-stage detectors* realizam a localização e classificação dos objetos separadamente para cada caixa delimitadora. Em contrapartida *one-stage detectors* realizam essas duas tarefas simultaneamente. Assim, somente uma execução na rede se demonstra necessária para realizar a detecção dos objetos na imagem. A arquitetura YOLO é um exemplo de *one-stage detectors* (TAN *et al.* 2020).

De acordo com Bochkovskiy *et al.* (2020), a YOLO *head* é segunda parte da arquitetura, sendo construída a partir das camadas desenvolvidas pelo *backbone,* esta desempenha um papel fundamental para a extração de características juntamente com a previsão realizada na inferência exercida durante as tarefas de detecção de objetos. Esta parte da arquitetura foi estruturada na versão YOLOv3 e então consolidada na versão YOLOv4, e por conseguinte, permanecendo até a versão atual. Esta estrutura é utilizada inicialmente para a detecção dos objetos desejados, incorporando etapas de detecção baseadas em âncoras e três níveis de granularidade. Redmon *et al*. (2018) demonstra que as detecções de objetos baseadas em âncoras se caracterizam por ser um método de detecção de objetos o qual utiliza *deep learning* e *bounding-boxes* pré-definidas (denominadas âncoras), como propostas para realizar a detecção. A ideia por traz de uma detecção de objetos utilizando âncoras consiste em referenciar e prever estas *bounding-boxes*. Portanto, cada objeto pode possuir um *label* caracterizando a ancora em questão, e cada imagem pode ter ou não a presença destes, sendo que podem existir classes distintas de *labels/ancoras.*

De maneira geral, o modelo YOLO mostra-se um sistema de detecção de objetos altamente eficiente e preciso e que tem sido fortemente aderido na comunidade de visão computacional. Sua capacidade demonstra-se o *state-of-the-art* em realizar a detecção de objetos, principalmente em tempo real. Em contrapartida, mesmo que o modelo apresente grandes índices de performance, ainda é possível perceber *bootlenecks* quanto a questão do *hardware* a ser utilizado. Assim, a disponibilidade de GPUs melhores, possibilitariam resultados mais rápido e eficientes. Além disso, a incorporação de novas técnicas, e mecanismos de atenção na arquitetura YOLO poderia melhorar sua *precision* e desempenho. Por fim, vale ressaltar a importância do *dataset* a ser utilizado. Embora que o modelo venha a apresentar ótimos resultados com bases de dados pequenas, este ainda possui a necessidade de grandes quantidades de imagens para poder apresentar bons números.

2.3 TRABALHOS CORRELATOS

Para realizar a busca por trabalhos correlatos, foram utilizados os portais: Google Scholar, Microsoft Research e Portal CAPES. Os termos de pesquisa escolhidos foram: *deep learning, remote sensing, landslides*. A plataforma Google Scholar retornou 17,100 resultados, a plataforma Microsoft Research retornou 11.615 resultados, e o Portal CAPES retornou 265 resultados. Para realizar a escolha dos artigos, foram considerados artigos publicados após 2020, que possuíam maior número de citações, e que apresentassem maior correlação com visão computacional, modelos de *deep learning* e landslides/deslizamentos. Ao final, foram selecionados 5 trabalhos que estão alinhados com esta pesquisa, sendo apresentados e discutidos na sequência.

De acordo com Ghorbanzadeh *et al.* (2021), visando a deteção de deslizamentos de terra, analisaram duas redes neurais convolucionais, sendo elas: U-Net e ResU-Net. A detecção é realizada utilizou dados disponíveis gratuitamente do satélite Sentinel-2, e do "ALOS digital elevation model". Os algoritmos foram treinados e avaliados usando dados de três áreas de estudo, sendo elas: Western Taitung County em Taiwan, Shuzheng Valley na China e Eastern Iburi no Japão. O modelo foi treinado usando um conjunto de dados do Japão e testado na área de teste de resistência da China, utilizando um tamanho de *patch* de amostra de 64x64 pixels. Segundo Ghorbanzadeh *et al.* (2021), as descobertas abrangem uma avaliação completa da transferibilidade em vários cenários de treinamento e teste nos três estudos de caso. Ao qual, o modelo ResU-Net apresentou o melhor *F1-Score*, atingindo 73,32%.

Em outro estudo, Ghorbanzadeh *et al.* (2022) investigaram a viabilidade de combinar um modelo de *deep learning* com análise de Object-Based-Image (OBIA) ao qual se baseia em regras utilizadas para categorizar e analisar objetos com imagens de satélite. Os autores utilizaram o modelo ResU-Net e as imagens são oriundas do satélite Sentinel-2. O modelo OBIA utiliza apenas cinco conjuntos de regras aplicadas no *dataset*, resultando em um mapa de calor referente a ResU-Net, sendo elas: tamanho, formato, textura, propriedades espectrais e informação contextual. Deste modo, foram avaliados três casos distintos: ResU-Net, OBIA e ResU-Net-OBIA. Para avaliar a *precision* dos modelos e dos mapas de detecção de deslizamentos calculou-se os valores de *precision*, *recall* e *F1-Score*. Segundo Ghorbanzadeh *et al.* (2022), a integração da estrutura ResU-Net-OBIA resultou no *F1-Score* de 76,56%.

Li *et al.* (2021) desenvolveram um *framework* baseado em *deep learning* visando realizar a detecção de deslizamentos de terra a partir de imagens hiperespectrais que possuem informações a partir de diversos comprimentos de onda no espectro eletromagnético. A

estrutura do algoritmo se divide em duas etapas: primeiramente, a *deep belief network* é utilizada para extrair as características espectrais-espaciais específicas para deslizamentos de terra. Posteriormente, as *features* e *constraints* de alto nível são incorporados a um classificador de regressão logística para verificar a presença de deslizamentos de terra. Segundo Li *et al.* (2021), o método alcançou uma *precision* de 97,91% para detecção de deslizamento de terra, enquanto outras abordagens, como máquina de vetores de suporte linear, divergência de informação espectral e correspondência de ângulo espectral alcançaram precisões de 94,36%, 84,50% e 86,44%, respectivamente.

Cheng *et al.* (2021) utilizaram o modelo You Only Look Once-Small Attention (YOLO-SA) para realizar a detecção de deslizamentos de terra em imagens de sensoriamento remoto por satélite. Para reduzir o número de parâmetros necessários no modelo, os autores propõem a utilização dos módulos residuais *group convolution* (GCONV) e *ghost bottleneck* (G-BNECK), substituindo os componentes de convolução e os módulos residuais baseados em convolução padrão. Complementarmente, um mecanismo de atenção é adicionado visando aumentar a *precision*. Cheng *et al.* (2021) compararam o YOLO-AS com 11 modelos distintos, sendo eles: Faster-RCNN, três tipos de EfficientDet, dois tipos de Centernet, SSD-eficiente e quatro tipos de modelos YOLOv4. Segundo os autores, os resultados apontam que o modelo YOLO-SA é mais eficiente, apresentando a pontuação *F1-Score* de 90,65%, taxa de omissão de 1.56% e taxa de erro de 16%.

Yi e Wanchang (2020) propuseram a utilização de *deep learning* para a detecção e mapeamento de deslizamentos de terra oriundos de terremotos a partir de imagens de satélite RapidEye. Os autores subdividiram o modelo em três etapas: inicialmente, é realizado o *preprocessing* de dados a serem treinados. Para aumentar a amostra de dados, são aplicadas várias transformações às imagens, aumentando artificialmente o tamanho do *dataset*. Posteriormente, uma rede de *deep learning* em cascata chamada LandsNet é construída. O LandsNet visa aprender diferentes recursos relacionados a deslizamentos de terra. Na etapa final, os mapas de deslizamento de terra identificados são processados usando processamento morfológico. Os resultados indicaram que a abordagem alcançou o melhor valor de *F1-Score* de aproximadamente 86,89%, sendo aproximadamente 7% e 8% maior que melhor que o *F1-Score* obtido pelos modelos ResUNet e DeepUnet, respectivamente. O Quadro 1 apresenta uma comparação entre as principais características dos trabalhos correlatos.

Quadro 1 – Comparação entre os trabalhos correlatos

Características / Trabalho	Base	Algoritmo	Resultado
Ghorbanzadeh *et al.* (2021)	Sentinel-2, "ALOS digital elevation model"	U-Net, ResU-Net	F1-Score: 73,32%
Ghorbanzadeh *et al.* (2022)	Sentinel-2	ResU-Net, OBIA, ResU-Net-OBIA	F1-Score: 76.56%
Li *et al.* (2021)	Imagens hiperespectrais	Deep belief network	Precision: 97,91%
Cheng *et al.* (2021)	Google Earth Pro	YOLO-SA	F1-Score: 90,65%, Omissão: 1.56%, Erro: 16%.
Yi e Wanchang (2020)	RapidEye	LandsNet	F1-Score: 86,89%

Fonte: elaborado pelo autor.

A partir do Quadro 1, pode-se observar que Ghorbanzadeh *et al.* (2021) utilizaram a rede neural convolucional U-Net e ResU-Net para a detecção de deslizamentos de terra. Os algoritmos foram treinados e testados em três áreas de estudo diferentes usando dados do satélite Sentinel-2 e modelos digitais de elevação. O ResU-Net obteve o melhor resultado, alcançando um *F1-Score* de 73,32%. Em outro estudo, Ghorbanzadeh *et al.* (2022) investigaram a combinação de um modelo de *deep learning* com análise de imagens baseada em objetos (OBIA) para detecção de deslizamentos de terra. A integração do ResU-Net com OBIA resultou em um *F1-Score* de 76,56%. Li *et al.* (2021) desenvolveram um *framework* baseado em *deep learning* para detecção de deslizamentos de terra em imagens hiperespectrais, alcançando uma *precision* de 97,91%. Cheng *et al.* (2021) utilizaram o modelo YOLO-SA para detecção de deslizamentos de terra em imagens de satélite, obtendo uma pontuação *F1-Score* de 90,65%. Yi e Wanchang (2020) propuseram o uso de *deep learning* para detecção e mapeamento de deslizamentos de terra em imagens de satélite RapidEye, alcançando um *F1-Score* aproximado de 86,89%.

Os trabalhos correlatos apresentam distintas técnicas de detecção de deslizamentos através de imagens de sensoriamento remoto, contudo nenhum testou modelos recentes como por exemplo o YOLOv8. Além disso, os trabalhos foram aplicados em sua maioria em território asiático o qual os deslizamentos se diferenciam consideravelmente dos presentes em Blumenau, principalmente devido a quantidade de vegetação e tipo do solo. Complementa-se que os trabalhos buscam detectar os deslizamentos em uma visão focada no deslizamento a ser detectado. Ou seja, não focam na análise das características espaciais nos derredores do deslizamento, e consequentemente perdendo muitos detalhes quando possíveis análises distintas sobre a região. Por fim, as detecções são realizas em imagens de baixo resolução, perdendo grande fidelidade de detalhes.

3 DESENVOLVIMENTO DO PROTÓTIPO

Neste capítulo será descrito as etapas de desenvolvimento do protótipo. Na seção 3.1 é detalhado os requisitos do trabalho, sendo eles: Requisitos funcionais (RF) e não funcionais (RNF). A seção 3.2 descreve o desenvolvimento e as técnicas e ferramentas utilizadas para cada etapa do ciclo de execução do protótipo. A seção 3.3 demonstra sobre a implementação do estudo realizado. A seção 3.4 descreve a utilização do YOLOv8. Por fim, a seção 3.5 apresenta e discute os resultados obtidos.

3.1 REQUISITOS

O protótipo foi desenvolvido atendendo os seguintes Requisitos Funcionais (RF) e Requisitos Não Funcionais (RNF):

a) permitir ao usuário carregar imagens a serem analisadas (RF);
b) demarcar deslizamentos e fissuras no solo utilizando as redes neurais convolucionais YOLOv8 (RF);
c) permitir ao usuário visualizar as detecções encontradas (RF);
d) ser desenvolvida utilizando a linguagem de programação Python (RNF);
e) ser capaz de realizar a análise em um tempo máximo de um minuto (RNF);
f) criar um *dataset* cronológicos dos deslizamentos através de imagens de sensoriamento remoto do CPRM e Google Earth (RNF);
g) aumentar o *dataset* utilizando técnicas de *preprocessing* e *data augmentation* (RNF);
h) utilizar a plataforma *Roboflow* como ferramenta de versionamento de dados (RNF).

3.2 TÉCNICAS E FERRAMENTAS UTILIZADAS

Nesta seção será descrito as ferramentas, técnicas e padrões utilizados na implementação do estudo em questão:

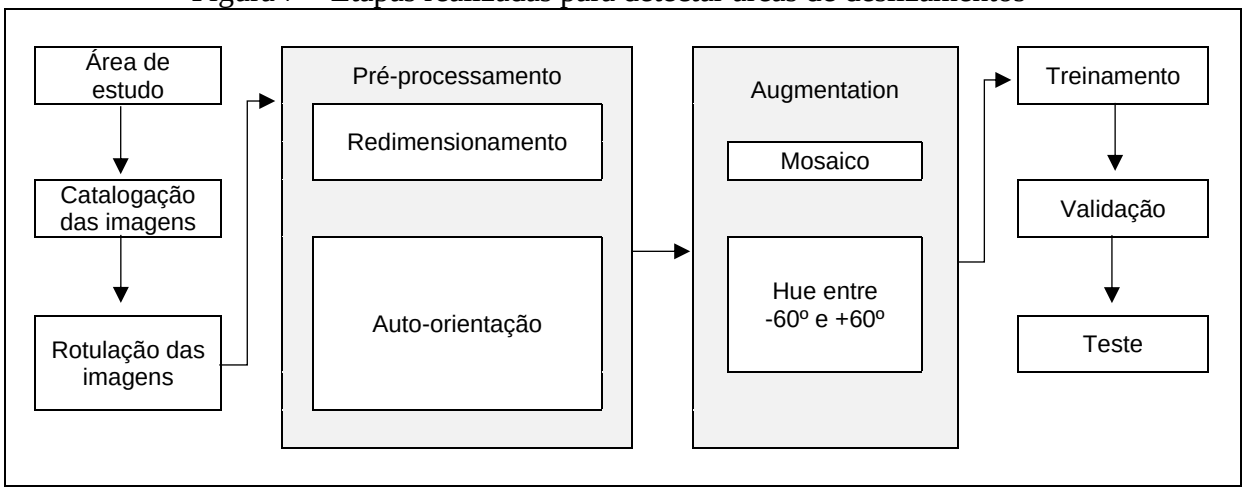
a) Python: a linguagem de programação Python foi utilizada na versão 3.10.6 localmente e 3.9.16 na ferramenta Google Colab;
b) Google Colab: ferramenta disponibilizada pelo Google para execução de arquivos. ipynb remotamente;
c) Jupyterlab: ferramenta disponibilizada pela empresa Jupyter para execução de arquivos .ipynb localmente;
d) Google Earth: utilizado para a coleta cronológica de imagens de satélite no município de Blumenau;

e) ArcGIS: utilizado com a versão 10.4 para a leitura de informações geográficas, e análise das camadas disponibilizadas pela defesa civil do município de Blumenau, com o intuito de localizar deslizamentos de terra pré-categorizados por especialistas da área;
f) Roboflow: plataforma online para criação e versionamento de *datasets* de imagens, juntamente com a possibilidade de gerar data *augmentation* e *preprocessing*;
g) YOLOv8: modelo de visão computacional escolhido para gerar o modelo e efetuar a inferência nas imagens;
h) Pytorch: utilizado na versão 1.13.1+cu116 remotamente e 2.0.0+cu117 localmente, é um framework de *machine learning* utilizado para realizar os cálculos necessários nos tensores exercidos no modelo de visão computacional;
i) WSL2: o subsistema Linux para Windows foi utilizado para montar o ambiente local e efetuar dos testes e análises realizados no estudo;
j) Ubuntu: utilizado na versão 22.04.1, é o subsistema GNU/Linux consumido localmente.

3.3 IMPLEMENTAÇÃO

Neste capítulo será descrito as etapas de implementação do protótipo. Na seção 3.3.1 é detalhado a área de estudo. A seção 3.3.2 aborda a construção do *dataset*. Por fim, a seção 3.3.3 demonstra a preparação do *dataset*. O diagrama de atividades, apresentado na Figura 7, indica as etapas realizadas para detectar áreas de deslizamentos a partir de imagens capturadas da cidade de Blumenau.

Figura 7 – Etapas realizadas para detectar áreas de deslizamentos



Fonte: elaborado pelo autor.

Primeiramente, definiu-se a região de estudo. Em seguida, iniciou-se a coleta das imagens da cidade de Blumenau, separando-as no intuito de facilitar a catalogação e o processamento dessas imagens através de algoritmo de visão computacional. Posteriormente, realizou-se a rotulação (*data labeling*), onde os objetos a serem detectados são identificados na imagem e demarcados. Definiu-se duas classes principais: Deslizamentos e Fissuras. Depois isso, efetua-se a etapa de redimensionamento (r*esizing*), normalizando o tamanho das imagens, e de *auto-orientation*, garantindo que todas as imagens possuam uma orientação padrão. A partir disso, efetua-se a *augmentation* gerando mosaicos para as imagens. Além disso, também se alterou o valor do Hue, gerando imagens com variação -60º e +60º, visando aumentar a variedade e riqueza do *dataset*. Com isso, torna-se possível efetuar o treinamento e validação do modelo YOLOv8. Por fim, são efetuados testes manuais para verificar a eficácia do modelo proposto.

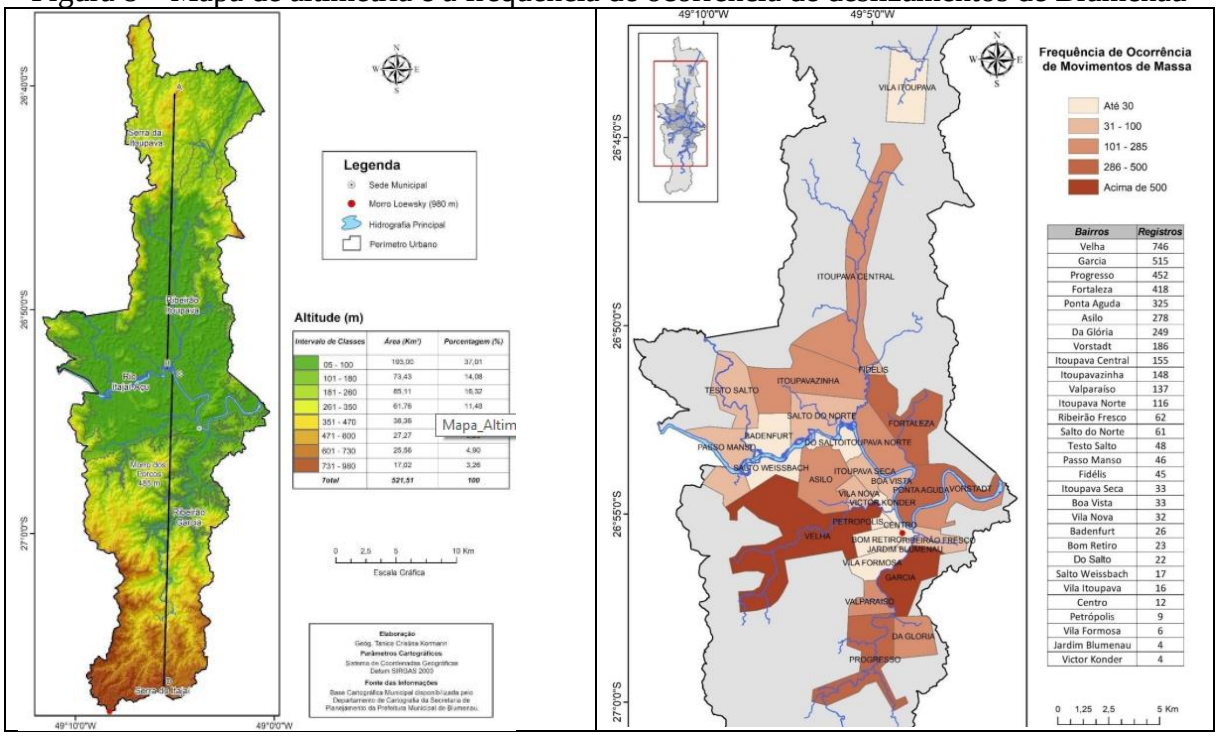
3.3.1 Área de estudo

A área selecionada para estudo engloba o município de Blumenau. De acordo com a Prefeitura de Blumenau, o município se localiza no nordeste de Santa Catarina e possui as coordenadas 26° ’55 10” Sul, 49° 03’ 58” Oeste, tendo a altitude de aproximadamente 21 metros acima do nível do mar. A área do município possuí 519,8 Km² sendo 206,8 Km² (39,78%) de área urbana e 313,0 Km² (60,22%) de área rural.

Kormann e Robaina (2020) caracterizam o relevo da área urbana de Blumenau a partir de parâmetros geomorfométricos de declividade, plano e perfil de curvatura e concluem que oito unidades de relevo no município possuem grande suscetibilidade à ocorrência de movimentos de massa tendo: declividade superior a 30%, perfil côncavo e plano convergente.

Segundo Kormann e Robaina (2020), a distribuição percentual das classes altimétricas indica o predomínio das altitudes de até 100m, correspondendo a 37,01% do território municipal. Esta faixa de altitude ocorre predominantemente na porção norte, enquanto as maiores altitudes ocorrem predominantemente no Vale do Ribeirão Garcia. Situado na porção sul, este vale apresenta as maiores amplitudes altimétricas em vales profundos em “V” esculpidos por uma drenagem encaixada. A Figura 8 apresenta a altimetria e a frequência de ocorrência de movimento de massa de Blumenau.

Figura 8 – Mapa de altimetria e a frequência de ocorrência de deslizamentos de Blumenau



Fonte: Kormann e Robaina (2020).

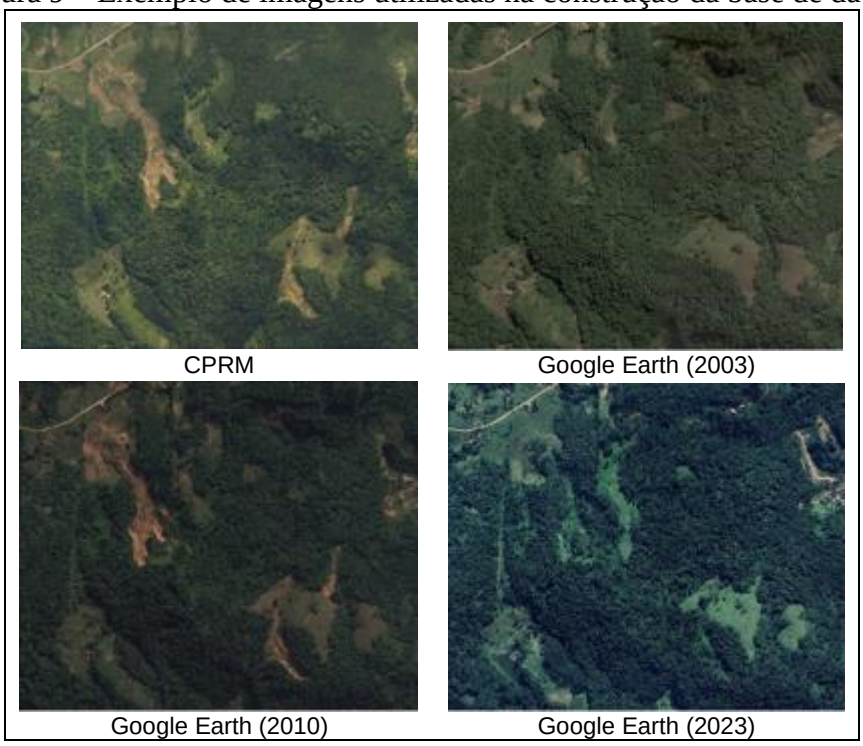
Ainda de acordo com Kormann e Robaina (2020), ao observar a distribuição espacial das ocorrências de deslizamentos atendidas pela Defesa Civil, pode-se observar os locais mais afetados ao longo do período compreendido entre os anos de 1997 e 2016. A representação cartográfica apresenta a frequência de ocorrência nos bairros de Blumenau, o que constitui um indicativo da intensidade e distribuição espacial do fenômeno. Neste sentido, nota-se que ao longo das duas décadas de registros, os três bairros que apresentaram maior frequência de ocorrências de movimentos de massa foram: Velha (17,66%); Garcia (12,19%) e Progresso (10,70%). Ou seja, correspondem a 40,55% de todos os registros de ocorrências atendidas, os três bairros que apresentam maior número de registros possuem, juntos, 23,74% da área do perímetro urbano municipal. Esta informação é indicativa da concentração espacial das ocorrências de deslizamentos, que se torna mais coerente ao considerar que estes bairros estão situados na porção sul do município.

3.3.2 Construção da base de dados

Para a construção do *dataset* foram utilizadas imagens aéreas disponibilizadas pelo Repositório Institucional de Geociências (CPRM) de 2015, possuindo escala de 1:50.000. A base possuí nitidez visual e consistência ao longo de todo o município. Em conjunto, foram utilizadas imagens ópticas de sensoriamento remoto obtidas através da ferramenta Google Earth. Estas imagens são datadas entre 2003 e 2023. Elas possuem entre 2.29 e 3.22 megapixels,

tendo média de 2.30 megapixels, e média de resolução de 2048x1121 pixels. A ferramenta ArcGis foi utilizada para localizar os deslizamentos. O Apêndice A exemplifica o processo de construção do *dataset* via ArcGis e no Google Earth. Destaca-se que todas os dados do CPRM e do Google Earth possuem três canais *Red, Green e Blue* (RGB). Os dados disponibilizados pelo CPRM foram abertos no ArcGis, ao qual foi possível visualizar os deslizamentos catalogados por geólogos da defesa civil. A Figura 9 apresenta uma comparação entre as imagens disponibilizadas pelo CPRM e pelo Google Earth de maneira cronológica.

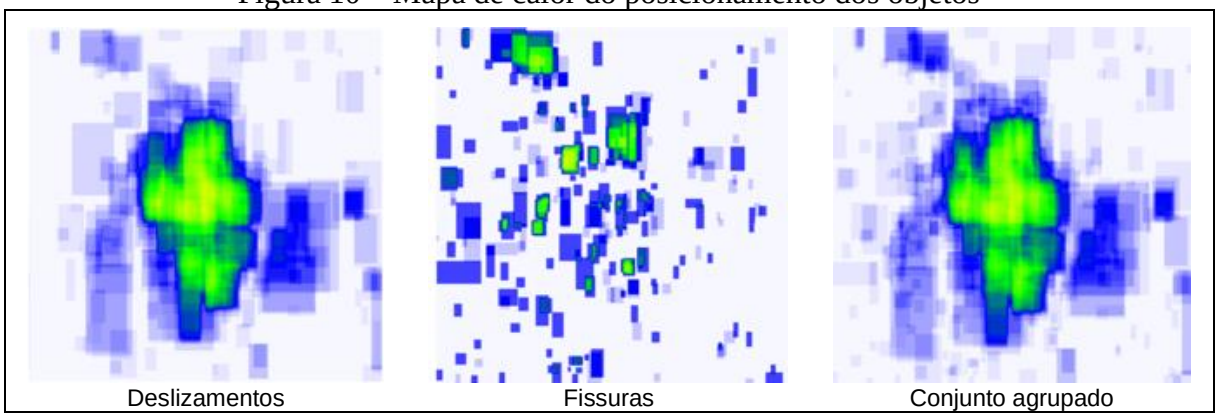
Figura 9 – Exemplo de imagens utilizadas na construção da base de dados



Fonte: elaborado pelo autor.

As imagens retiradas do Google Earth, também foram localizadas através da marcação disponibilizada no ArcGis. No caso do Google Earth, que possui cronologia, para cada deslizamento foram retiradas de 1 a 20 imagens, sendo em média uma por ano, contudo, nos casos em que houveram mais de uma imagem por ano, mais imagens foram selecionadas, principalmente quando houve sinais de artefatos que poderiam auxiliar na diversidade dos dados presentes no *dataset*, como nuvens e ruídos. Vale ressaltar que em alguns casos, certos anos não estavam disponíveis. Deste modo, foram ignorados. Ao final, foram catalogados 759 objetos, sendo estes 505 deslizamentos (66.53%) e 254 fissuras (33.47%). Ressalta-se que se optou por definir apenas estes tipos (deslizamentos/fissuras) devido a quantidade ocorrências catalogadas. Estes objetos estão distribuídos no total de 575 imagens onde 299 (52%) possuem deslizamentos, 140 (24.34%) possuem fissuras e 201 (34.96%) não possuem objetos. A Figura 10 exibe o posicionamento das ocorrências mapeadas através de um mapa de calor.

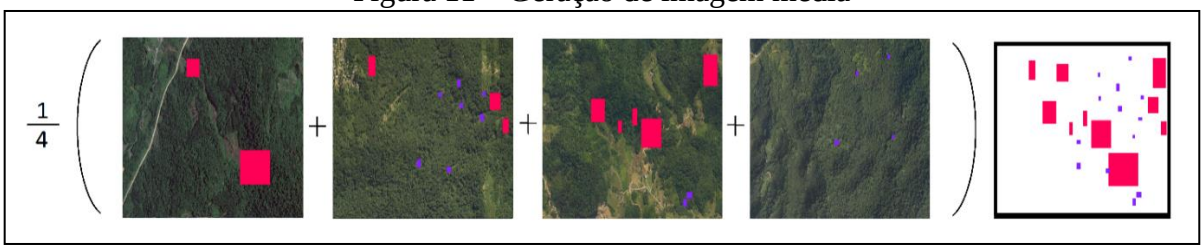
Figura 10 – Mapa de calor do posicionamento dos objetos



Fonte: elaborado pelo autor.

Para a construção do mapa de calor, criou-se uma imagem média a partir da diferença entre cada imagem, conforme equação: $[n \times M] = (\Phi 1, \Phi 2, \ldots, (\Phi)\ M)$. A partir dela, pode-se observar as características dos arredores dos locais onde ocorrem ou existem deslizamentos ou fissuras, auxiliando possivelmente na definição do tipo de ocorrência. A Figura 11 ilustra o processo da geração da imagem média.

Figura 11 – Geração de imagem média

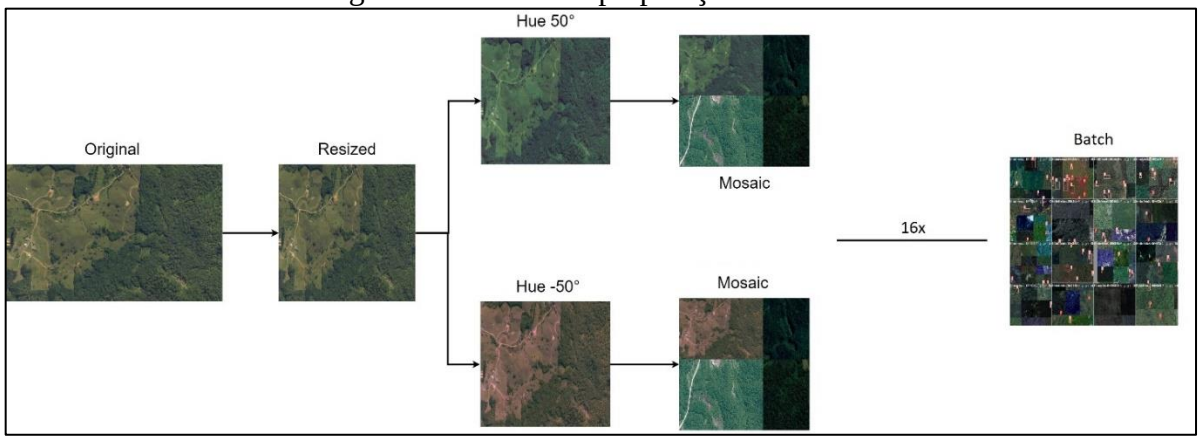


Fonte: elaborado pelo autor.

3.3.3 Preparação do *dataset*

A Figura 12 apresenta o diagrama de como foram realizadas as etapas de *preprocessing* e *augmentation*, resultando por fim em um *batch*. No campo da visão computacional e *machine learning*, o *batch* caracteriza-se como um hiperparâmetro o qual determina o número de amostras a serem processadas simultaneamente antes de se atualizar os parâmetros internos do modelo. Ao escolher-se o tamanho do *batch*, se deve analisar o impacto no processo de treino, pois este influencia consideravelmente a quantidade de recursos computacionais necessários para se efetuar o treino. Um tamanho de *batch* maior exige por consequência mais memória disponível, mas em contrapartida, efetua um treino mais rápido. Um *batch* no tamanho de 16 ou 32 geralmente é considerado adequado para a maioria das tarefas de visão computacional.

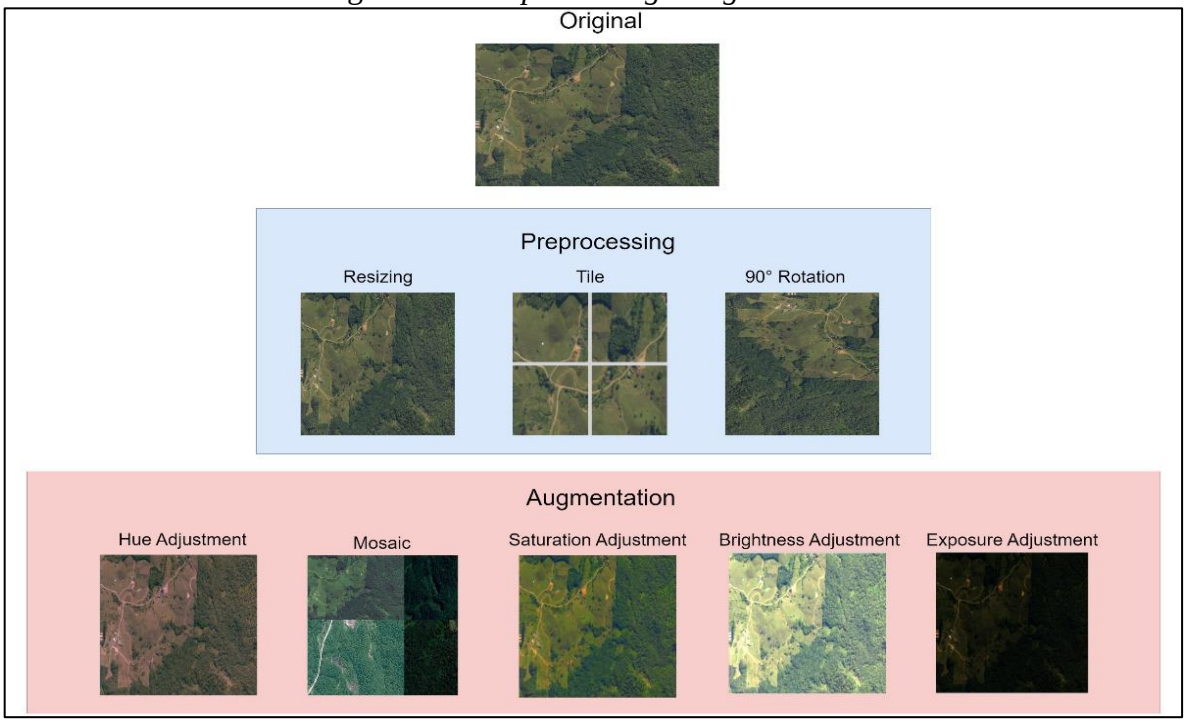
Figura 12 – Fluxo de preparação do *dataset*



Fonte: elaborado pelo autor.

Inicialmente, os rótulos referentes as *bouding-boxes* foram armazenados em arquivos de texto (.txt) no padrão exigido pela arquitetura YOLOv8. A Plataforma Roboflow, foi utilizada para fazer o versionamento do *dataset* e para: (i) criar a rotulação dos objetos, (ii) catalogação e estudo dos dados, (iii) análise da saúde do *dataset* e para (iv) possibilitar a execução das etapas de *preprocessing* e *data augmentation*. As etapas de *preprocessing* testadas foram de: *resizing* 640x640, *resizing* 800x800, *resizing* 1024x1024, *Tile* 3x3, *Tile* 2x2, 90° *rotation* e *null filtering*. As etapas de *augmentation* testadas foram: *mosaic* 3x, *mosaic* 1x, h*ue adjustment*, *saturation adjustment*, *brightness adjustment* e *exposure adjustment*. A Figura 13 ilustra os procedimentos *preprocessing* e *augmentation*. No Apêndice B, encontra-se o detalhamento das funcionalidades do Roboflow.

Figura 13 – *Preprocessing* e *augmentation*



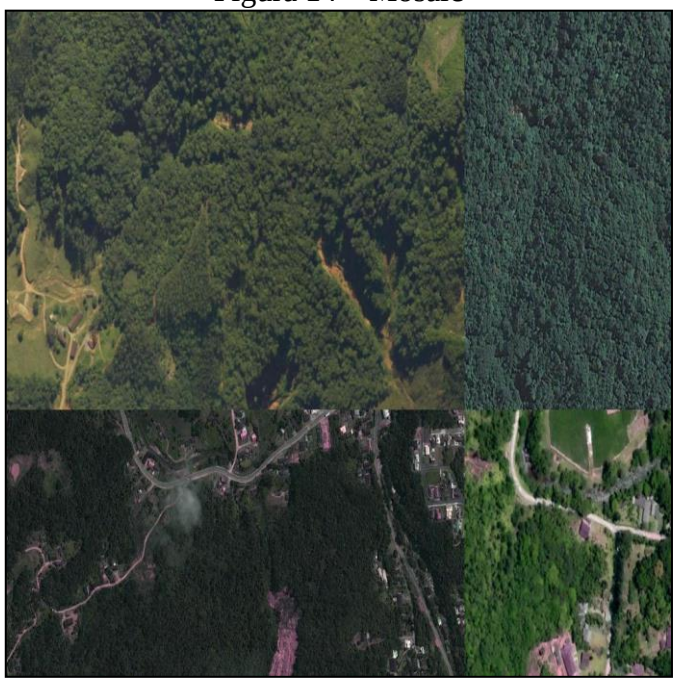
Fonte: elaborado pelo autor.

Normalmente, durante o treino de um modelo de visão computacional, opta-se por fazer *resizing* de todas as imagens, normalizando-as em 640x640, como por exemplo os próprios modelos do YOLO disponibilizados pela Ultralytics (YOLOv8s) ou pelo modelo Microsoft COCO (LIN *et al.* 2014), o qual possui resolução de 640×480. Contudo, no modelo treinado observou-se que esta resolução se demonstra insuficiente para avaliar a complexidade de imagens de sensoriamento remoto. Deste modo, foram realizados teste com as resoluções de 640x640, 800x800 e 1024x1024. Destaca-se que resoluções maiores que as citadas não foram testadas por requirir processamento computacional elevado para o *setup* experimental. Deste modo, seria necessário reduzir o *batch size*, comprometendo os resultados do experimento. Notou-se que a resolução de 640x640 teve um aumento em sua *precision* de 14% quanto a de 800x800 e de 27% quanto a de 1024x1024. Para as taxas de *recall*, se teve um aumento de 0.33% quanto a de 800x800 e de 11% quanto a de 1024x1024 no *recall*. Os dados se demonstram expressivos principalmente quanto as fissuras, estas por serem objetos menores, e por consequência, com menor riqueza de detalhes.

Na etapa de *preprocessing*, utilizou-se a técnica de *auto-orienting*, normalizando o *dataset*, assim como prevenindo a possibilidade de *mismatch* entre anotação e imagem. Além disso, outros tipos de *data-augmentation* foram testados, como por exemplo a segmentação das imagens em *tiles* 3x3 e 2x2. Contudo, neste caso, os resultados não se demonstraram satisfatórios, mantendo a *precision* e *recall* sempre muito próximas, e com pouca melhoria em relação ao *ground truth*. As imagens também foram rotacionadas em 90°: *Clockwise*, *Counter-Clockwise*, *Upside Down*, mas estes também acabaram por demonstrar *overfitting*. Por fim, a filtragem de nulos foi testada, visto que 34.96% das imagens não possuem objetos, deste modo com somente 25%, contudo os resultados demonstram-se piores que os do Ground Truth.

Na etapa de *augmentation*, utilizou-se a técnica de criação de mosaico, criada por Glenn Jocher (BOCHKOVSKIY *et al.* 2020) e lançada pela primeira vez no YOLO v4. Esta técnica funciona combinando quatro imagens em uma. Nos testes realizados, obteve-se melhoras significativas nos resultados, principalmente devido ao fato de ter sido aumentado a diversidade do *dataset*, melhorando a *precision* e *recall*. A Figura 14 demonstra um exemplo da utilização da técnica de *mosaic*.

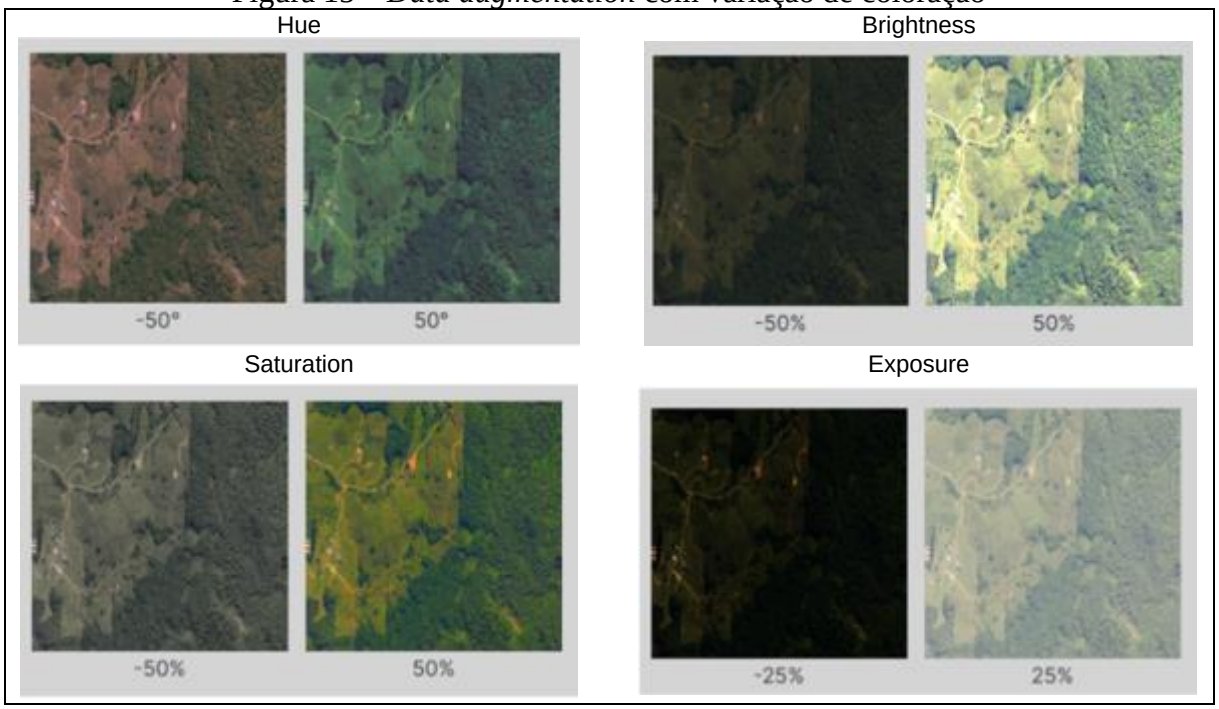
Figura 14 – Mosaic



Fonte: elaborado pelo autor.

Em outro teste, analisou-se qual processamento normalizaria melhor a coloração das imagens, principalmente devido a variação presente nos dados retirados da plataforma Google Earth. Em resumo, o processamento que demonstrou melhor resultado foi a variação de Hue entre 60° e +60°. As demais correções de saturação, brilho e exposição também vieram a apresentar bons resultados, mas inferiores aos apresentados pela variação de Hue. A Figura 15 demonstra exemplos dos tipos de *data augmentation*.

Figura 15 – *Data augmentation* com variação de coloração



Fonte: elaborado pelo autor.

3.4 UTILIZAÇÃO DO MODELO YOLO

Para realizar o treino do modelo YOLOv8, inicialmente é necessário importar a pacote `ultralytics` disponibilizada pela empresa Ultralytics através do sistema de gerenciamento de pacotes `pip` para executar os comandos de treino e inferência. Além disso, para facilitar o versionamento do *dataset* também é necessário importar o pacote `roboflow`. O Quadro 2 exemplifica a instalação de ambos os pacotes.

Quadro 2 – Instalação do modelo YOLOv8

```
1  pip install ultralytics
2  pip install roboflow
```

Fonte: elaborado pelo autor.

Posteriormente, aconselha-se executar o comando de checagem da biblioteca Ultralytics para garantir que a placa de vídeo foi reconhecida e que os devidos componentes estão de fato instalados. O Quadro 3 exemplifica a execução da checagem e o Quadro 4 demonstra um possível *output*, este pode variar de acordo com as configurações do ambiente de execução.

Quadro 3 – Teste do modelo YOLOv8 e suas dependências

```
1  ultralytics.check()
```

Fonte: elaborado pelo autor.

Quadro 4 – *Output* do comando de teste do modelo

```
1  Ultralytics YOLOv8.0.121 Python-3.10.12 torch-2.0.1+cu118 CUDA:0
   (Tesla T4, 15102MiB)
2  Setup complete 2 CPUs, 12.7 GB RAM, 24.1/78.2 GB disk)
```

Fonte: elaborado pelo autor.

Posteriormente, pode-se efetuar o *download* do *dataset* diretamente do site Roboflow, conforme demonstra o código do Quadro 5.

Quadro 5 – *Download* do *dataset* do Roboflow

```
1  Rf = Roboflow(api_key=”apy_key”)
2  project = rf.workspace("deslizamentos").project("landslide-test")
3  dataset = project.version(3).download("yolov8")
```

Fonte: elaborado pelo autor.

A partir disso é possível realizar o treino do modelo utilizando apenas linha de comando. Ressalta-se os argumentos: *task*¸ o qual determina que será realizado o treino de um modelo de detecção e não de classificação, *mode* que determina se será realizado treinamento e não inferência, *model*, o qual determina o modelo da YOLOv8 que será utilizado, *data*, que determina o caminho do arquivo `.yaml` que detêm os diretórios dos dados de treino, validação e teste, *epochs*, definindo o número de *epochs* máxima do treino em questão, *patience*, informando o número de *epochs* em que o modelo treinara continuará treinando em caso de não se obterem melhorias no modelo e *imgsz* onde se determina a resolução da imagem. O Quadro 6 exemplifica uma execução do comando de treino, e o Quadro 7 exemplifica o *output* da execução.

Quadro 6 – Execução do treino do modelo

```
Yolo task=detect mode=train model=yolov8s.pt data=/content/Landslide-Detection-Augmentation-3/data.yaml epochs=10 patience=500 imgsz=1024
```

Fonte: elaborado pelo autor.

Quadro 7 – Output da execução do treino do modelo

```
10 epochs completed in 0.149 hours.
Model summary (fused): 168 layers, 11126358 parameters, 0 gradients
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 4/4
[00:06<00:00,  1.61s/it]
                   all        115        162      0.415      0.412      0.364      0.171
               Fissure        115         60      0.522      0.167      0.272      0.127
             Landslide        115        102      0.307      0.657      0.455      0.214
Speed: 7.0ms preprocess, 7.8ms inference, 0.0ms loss, 3.2ms postprocess per image
```

Fonte: elaborado pelo autor.

Com o modelo treinado, é possível realizar a inferência das imagens de teste. Ressalta-se os argumentos: *task*¸ o qual determina que será realizado a inferência nas imagens e não o treino do modelo, *model*, que demonstra o caminho para o modelo treinado, *conf* que informa o *threshold* de confiabilidade das detecções, *source*, denominando o caminho das imagens a serem inferidas e *save*¸ deliberando que dados serão persistidos. O Quadro 8 exemplifica uma execução do comando de inferência, e o Quadro 9 apresenta seu *output*. A Figura 16 demonstra alguns resultados de inferência do comando executado, no qual pode-se observar a detecção de deslizamentos e fissuras presentes em três imagens distintas.

Quadro 8 – Inferência de imagens para detecção de deslizamentos e fissuras

```
yolo task=detect mode=predict model=/lddetection/LDDetectio/runs/detect/train/weights/best.pt conf=0.25 source=/lddetection/LDDetection/test/images save=True
```

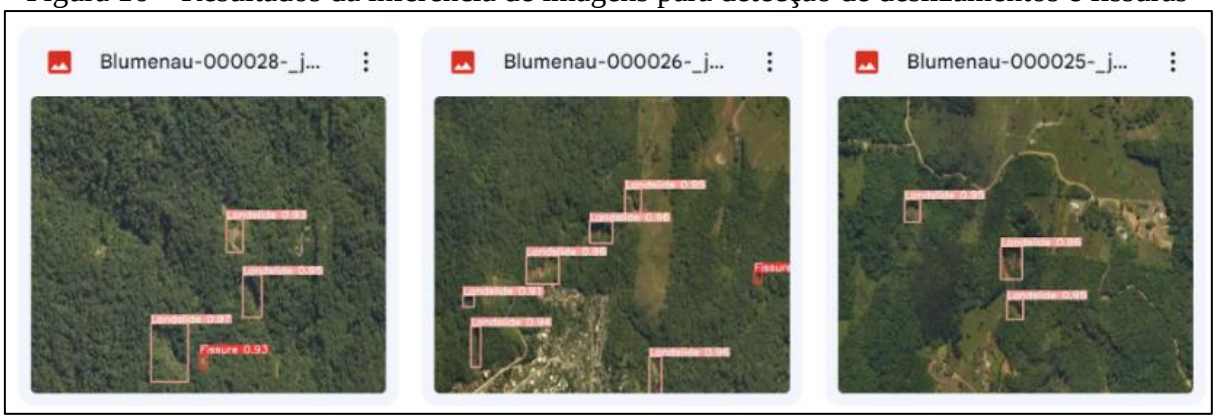
Fonte: elaborado pelo autor.

Quadro 9 – Output da inferência

```
image 1/1 /content/Tests/images/Blumenau-000001.jpg: 1024x1024 1 Landslide, 32.6ms
image 2/2 /content/Tests/images/Blumenau-000002.jpg: 1024x1024 2 Fissure, 3 Landslide, 32.6ms
image 3/3 /content/Tests/images/Blumenau-000003.jpg: 1024x1024 (no detections), 32.6ms
Speed: 6.0ms preprocess, 23.5ms inference, 2.6ms postprocess per image at shape (1, 3, 1024, 1024)
```

Fonte: elaborado pelo autor.

Figura 16 – Resultados da inferência de imagens para detecção de deslizamentos e fissuras



Fonte: elaborado pelo autor.

3.5 RESULTADOS E DISCUSSÕES

Os experimentos foram realizados a partir da linguagem Python-3.9.16, torch-1.13.1+cu116 CUDA:0 (Tesla T4, 15102MiB) e localmente utilizando Python-3.10.6, torch-2.0.0+cu117 CUDA:0 (NVIDIA GeForce RTX 3080, 10240MiB) tendo um subsistema GNU/Linux Ubunto 22.04.1 com WSL2 em uma plataforma Windows 11 como sistema operacional. Os treinos do modelo foram realizados com *batch size* 16, utilizando 8 *workers*. Após a criação do *dataset*, foram testados vários processos de *preprocessing* (*resizing*, *tile creation*, *rotation* e *null Filtering*) e *data augmentation* (*mosaic creation*, *saturation adjusment*, *brightness adjusment* e *exposure adjusment*) selecionando apenas os *preprocessing* e augmentation que obtiverem o melhor resultado, com o intuito de calibrar o *dataset* e evitar *overfitting* dos dados, gerando imprecisão e debilitando a performance do modelo. Após esta etapa, totalizaram-se 954 imagens de treino (70%), 273 de validação (20%) e 136 de teste (10%). As imagens de treinamento foram utilizadas para treinar o modelo, as de validação para gerar os dados de *recall* e *precision* diretamente pelo modelo YOLOv8, e de teste para reservar imagens para serem avaliadas manualmente.

O Quadro 10 apresenta os resultados dos testes de *preprocessing* e *augmentation*. Os testes de *resizing* foram feitos com somente a técnica de *resizing* juntamente com a técnica de *auto-orienting*. Esta se demonstra um requisito para o funcionamento eficiente do modelo YOLOv8, deste modo, foi aplicada em todos os testes. Os testes de *augmentation* foram feitos somente com a técnica escolhida em questão, juntamente com *auto-orienting.*

Quadro 10 – Resultados dos testes de *preprocessing* e *augmentation*

Phase	Step	Precision	Recall	mAP50	mAP50-95
Resize	600x600	0.73	0.446	0.514	0.244
	800x800	0.639	0.447	0.506	0.256
	1024x1024	0.572	0.501	0.528	0.266
Tile	3x3	0.447	0.380	0.329	0.151
	2x2	0.503	0.451	0.428	0.198
Rotate	90°	0.745	0.380	0.47	0.231
Null filtering	-	0.647	0.420	0.506	0.255
Mosaic	3x	0.978	0.851	0.929	0.833
	1x	0.935	0.672	0.778	0.618
Adjustment	Hue	0.633	0.403	0.476	0.241
	Saturation	0.618	0.437	0.497	0.246
	Brightness	0.568	0.454	0.499	0.247
	Exposure	0.533	0.506	0.504	0.244

Fonte: elaborado pelo autor.

Considerando os resultados individuais do Quadro 10, foram realizados experimentos agrupando os tipos de *preprocessing* e *augmentation*. Neste caso, foram selecionados os *augmentations* que obtiveram melhores resultados: *hue adjustment* e *mosaic creation*, além de

aplicar o *resizing* de 1024x1024 e efetuar a *auto-orienting* em todas as Imagens. A escolha de somente estas *augmentations* é oriunda do objetivo de eliminar possíveis *overfitting*.

Para reforçar os resultados obtidos na fase de treinamento e validação, aplicou-se algumas métricas durante a compilação e somatória dos pesos da rede neural, sendo as métricas denominadas, *precision*, *recall*, *F1-Score*, mAP50 e mAP50-95. Segundo Goutte e Gaussier (2005), *precision* que denomina a proporção de retornos denominados como corretos pelo sistema, esta denomina o quanto que o sistema é capaz de detectar objetos corretamente, deste modo, *precision* altas apresentam melhores resultados sendo que uma *precision* baixa qualifica um sistema com poucas detecções. A métrica e *recall*, determinando a proporção de entidades que o sistema de fato retorna, esta é importante para avaliar a frequência de falsas detecções presentes, em síntese, quanto menor o *recall* maior tende a ser a *precision*, sendo que modelos com *recall* alto tendem a possuir muitos objetos erroneamente classificados. Utiliza-se a métrica F-score, que apresenta um meio harmônico entre a *precision* (P) e *recall* (R), esta é útil para averiguar a constância do modelo, garantindo que não haja picos de *recall* ou *precision*, denominando que estes avancem somente de maneira proporcional. Por fim, a *Mean Average Precision* (mAP) caracteriza a precisão média em determinados blocos, sendo eles as últimas 50 e 95 epochs. A mAP tem como papel avaliar a proporção em que a precisão aumenta ou diminui em relação ao recall e vice e versa. Esta se demonstra útil para averiguar a constância ou evolução do modelo, sendo que tanto valores muito altos como muito baixos apresentam imprecisão nas detecções.

A média da *precision* das últimas 50 e 95 epochs é o principal índice utilizado para avaliar a performance geral de um algoritmo de detecção de objetos (CHENG *et al.* 2021). Inicialmente, definiu-se 500 epochs, contudo, os resultados tendem a alcançar um estado de platô entre 250 e 300 *epochs*. A *precision* ficou em 0.959 para o conjunto, 0.943 para fissuras e 0.974 para deslizamentos. O *recall* teve 0.787 para o conjunto, 0.697 para fissuras e 0.877 para deslizamentos. A mAP das últimas 50 epochs teve o resultado de 0.863 para o conjunto, 0.802 para fissuras e 0.924 para deslizamentos. Por fim, a mAP das últimas 95 epochs teve o resultado de 0.759 para o conjunto, 0.659 para fissuras e 0.86 para deslizamentos. Deste modo, ao avaliar 273 imagens, a *precision* apresentou-se alta em 145 instâncias de fissuras, e 261 instâncias de deslizamentos, isso caracteriza que o modelo está conseguindo detectar grande parte dos objetos separados para validação. O *recall* demonstra que foi possível prevenir um número grande de falsas detecções de deslizamentos, principalmente pela simplicidade deste tipo de objeto, que normalmente é menor e com menos variação de textura. Por outro lado, o *recall* para a detecção de deslizamento foi um pouco maior em relação a detecção de fissuras,

isto é, devido ao fato de os objetos possuírem uma variação maior quanto as suas características morfológicas. Portanto, para melhorar este índice seria necessário construir um *dataset* mais vasto, englobando um maior número de deslizamentos com características diversas. Por fim, a relação entre mAP50 e mAP50-95 consolida a análise em que são necessárias mais imagens de deslizamentos, pois não apresentou melhora quando comparado com a mAP. O Quadro 11 apresenta os resultados do modelo proposto, após selecionar somente os *preprocessings* e *augmentations* desejados.

Quadro 11 – Resultados obtidos pelo modelo

Classe	Imagens	Instances	Precision	Recall	mAP50	mAP50-95
Fissure	273	145	0.943	0.697	0.802	0.659
Landslide	273	261	0.974	0.877	0.924	0.86
Total	273	406	0.959	0.787	0.863	0.759

Fonte: elaborado pelo autor.

Para realizar a análise manual/visual, executou-se o algoritmo de inferência com o índice de confiabilidade de 0.25. Optou-se por este valor pois ele apresentou melhores resultados ao filtrar falsos positivos / negativos, assim como evita o aumento da taxa de rejeição de casos positivos. A taxa de *overlap* foi mantida no padrão de 40%, tal qual demonstrou poucos casos de detecções múltiplas com presença de *overlapping*. A Figura 17 exibe a comparação entre uma demarcação manual e a detecção do modelo desenvolvido.

Figura 17 – Comparação do resultado com o *ground truth*

Detecção YoloV8 Ground Truth

Fonte: elaborado pelo autor.

A partir da Figura 17 pode-se observar que o modelo apresentou resultados significativos quanto a detecção de deslizamentos de terra através de imagens de sensoriamento remoto. Foi possível realizar a detecção da maior parte dos objetos demarcados. Os itens que não foram detectados em sua maioria apresentam, densa vegetação “a” ou omissão de detalhes “b”. Este tipo de deslizamento apresenta dificuldades para a catalogação manual, o que em certos aspectos explica os problemas apresentados pelo modelo. Complementa-se que o modelo possui taxas muito baixas de falsos positivos, mesmo em casos em que os falsos objetos se

assemelham bastante aos deslizamentos. A Figura 18 apresenta a matriz de confusão dos resultados obtidos.

Figura 18 – Matriz de confusão

Fissure	0.70	0.00	0.50
Landslide	0.02	0.88	0.50
Background	0.28	0.12	0.00
	Fissure	Landslide	Background

Fonte: elaborado pelo autor.

A matriz de confusão obtida demonstra que 70% das fissuras foram detectadas corretamente (TP), 2% foram detectadas como deslizamento (FP) e 28% não foram detectados (FN). Quanto a detecção de deslizamentos, 88% foram detectados corretamente, nenhum deslizamento foi detectado como fissura (FP), e 12% não foram detectados.

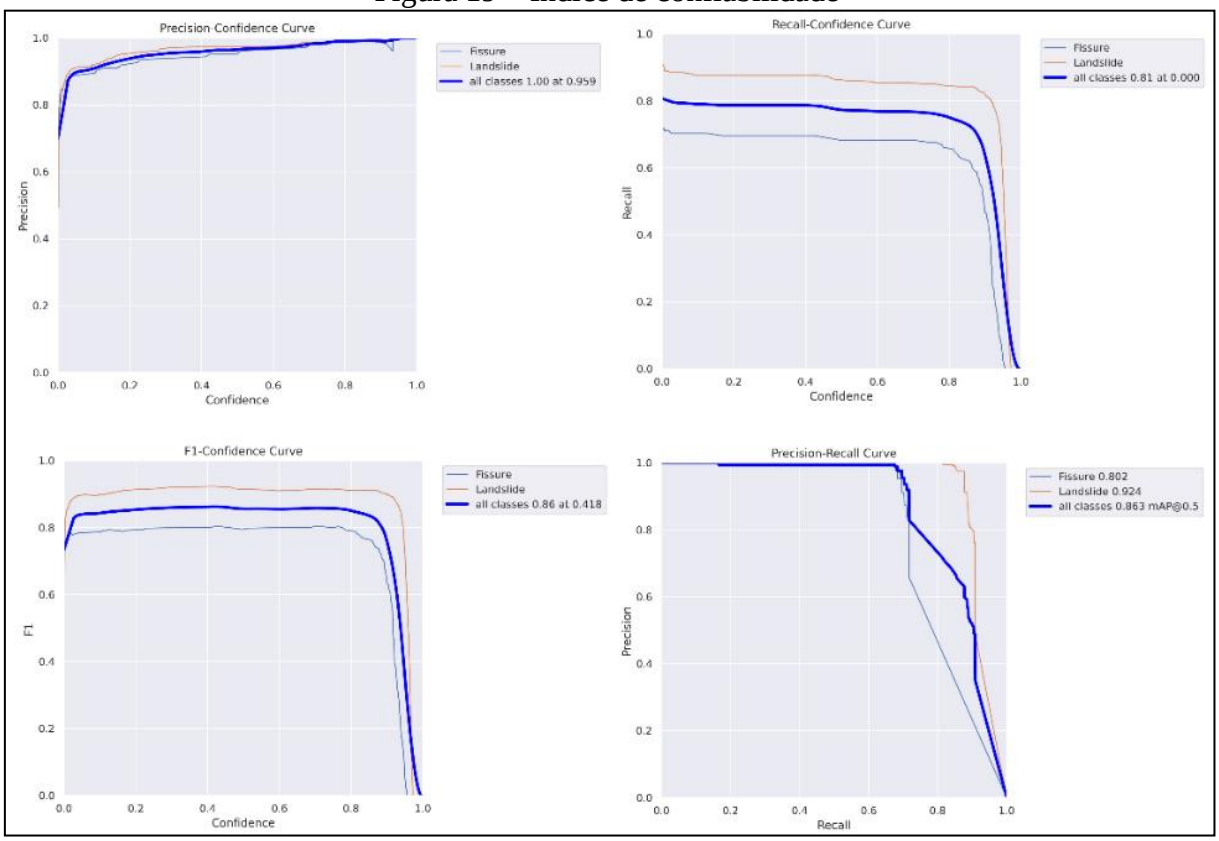
A partir do gráfico de *precision-confidence* da Figura 19, é possível perceber que o modelo manteve o índice de confiabilidade em um nível alto durante toda a sua execução. Com *confidence* 0, a *precision* ficou em torno de 0.7, alcançando então rapidamente a *precision* de aproximadamente 0.82. Posteriormente, a *precision-confidence* do modelo foi gradativamente aumentando até alcançar seu máximo com a *confidence* em aproximadamente 0.75, caracterizando um aumento satisfatório quanto a evolução dos índices de confiabilidade.

De acordo com o gráfico de *recall-confidence* é possível validar a confiabilidade do modelo. Em síntese, percebe-se que a detecção de falsos positivos se manteve estável ao longo do treino, não possuindo aumentos bruscos quando correlacionados com o aumento da *precision* demonstrado pelo gráfico *precision-confidence.* Isso acontece pois a confiança se estabiliza em aproximadamente 0.9.

A acurácia e confiabilidade pode ser avaliada com o gráfico de F1-Confidence, este demonstrou um platô com F1-*Score* 0.8 entre aproximadamente 0.01 e 0.9 de *confidence*. Portanto, ao correlacionar este fator com os gráficos de *precision-confidence* e *recall-confidence*, conclui-se que não houve picos de deterioração quanto a confiança e qualidade do modelo.

Por fim, o gráfico de *precision-recall* consolida os demais gráficos ao demonstrar que houve *precision* absoluta com *recall* menor que aproximadamente 0.7. Deste modo, a *precision* não se alteraria ao diminuir o recall, não sendo possível realizar a troca entre detectar mais TPs em troca de detectar mais FP.

Figura 19 – Índice de confiabilidade



Fonte: elaborado pelo autor.

A partir da Figura 19, também pode-se observar que o índice de confiabilidade médio, ao longo de todas as detecções, se mostrou consideravelmente alto, sendo que em todas as classes, o média de confiabilidade foi de 0.959 em 1.000 de *precision*, e de 0.810 em 0.000 de *recall*. Em relação de *F1-Score*, manteve o valor médio de 0.86. Por fim, o valor da curva de *precision-recall* foi de 0.802 para fissuras, 0.924 para deslizamentos, sendo que em todas as classes, a média mAP@0.5 ficou em 0.863.

4 CONCLUSÕES

Desastres ambientais são ocorrências que afetam milhares de pessoas no município de Blumenau e do Médio Vale do Itajaí. Possuir os mecanismos necessário para prevenir-se contra novos desastres e tomar as medidas governamentais e ambientais necessárias é imprescindível para o planejamento urbano-social do município. A falta de tecnologias as quais possibilitam a detecção de deslizamentos, de maneira automatizada, afeta a rapidez e o dinamismo do trabalho dos profissionais responsáveis pelo bem-estar da população. Desta maneira, estudos de novas formas de se realizar a detecção ou monitoramento destes eventos são de grande interesse dos órgãos públicos.

Neste sentido, desenvolveu-se um protótipo que utiliza imagens de sensoriamento remoto do município de Blumenau. Foram catalogadas 100 imagens do CPRM, obtidas por meio do software ArcGIS e 475 imagens coletadas pelo Google Earth para criar o conjunto de dados, totalizando 759 objetos, sendo 505 de deslizamentos (66.53%) e 254 de fissuras (33.47%). Além disso, 201 imagens (34.96%) que não possuem objetos foram utilizadas para aumentar a acurácia do modelo quanto a falsas detecções. Em seguida, foram aplicadas técnicas de *preprocessing* e *data augmentation* para complementar a amplitude do *dataset*. Essas etapas consistiram em normalizar as imagens em um padrão específico, criando variações (*auto-orient, resize, mosaic* e *hul adjustment*) visando abranger um número maior de tipos de deslizamento. Em suma, resultaram-se em um total de 954 imagens usadas para treinamento e validação. Para detectar áreas com deslizamentos, utilizou-se o modelo YOLOv8, aproveitando um modelo pré-treinado do conjunto de dados Microsoft COCO para gerar um modelo específico para este tipo de detecção. O modelo YOLOv8 utilizado foi desenvolvido com a linguagem de programação Python, versão 3.7, utilizando a biblioteca Keras juntamente com Pytorch. Para sua criação, os objetos presentes nas imagens foram divididos entre *deslizamento* e *fissura*, pois devido as diferenças visuais, juntamente com as de sua natureza, a detecção por parte do modelo se demonstrava inviável.

Os resultados alcançados se aproximam do atual *state-of-the-art* quando se trata do uso de visão computacional para detecção de deslizamentos de terra. A utilização de um modelo recente do YOLO possibilitou avanços quanto aos trabalhos correlatos, obtendo *F1-Score* de 0.86, enquanto em nos trabalhos correlatos a pontuação foi de 0.73, 0.73, 0.86, sendo pior apenas que o trabalho desenvolvido por Cheng *et al.* (2021) que apresentou *F1-Score* de 0.90, contudo, este aplicou o modelo YOLO-SA, o qual utiliza um modelo de atenção, sendo um paradigma completamente distinto do desenvolvido com redes neurais convolucionais. A

comparação entre a detecção realizada pelo modelo e o *ground-truth* demonstrou-se muito próxima, portanto, é possível afirmar que o modelo treinado apresenta grande fidelidade de detecção, principalmente quando comparado com os dados providos pela matriz de confusão apresentada. Grande parte dos problemas apresentados durante a detecção são devidos a vegetação densa ou omissão de detalhes. Desta forma, é possível afirmar que aumentando o número de imagens no *dataset* será possível melhorar a precisão do modelo.

Ressalta-se também que os resultados obtidos também caracterizam a detecção de deslizamentos em imagens de alta resolução, possibilitando uma análise macro das áreas afetadas. Assim sendo, características morfológicas dos locais podem ser analisadas por especialistas, o que apresenta um grande avanço em relação aos trabalhos correlatos, os quais focavam-se em detectar deslizamentos com resoluções menores, e com foco específico no objeto. Por fim, conclui-se sobre a importância da normalização dos dados em um modelo de visão computacional com o *dataset* oriundo de imagens providas por sensoriamento remoto. Visto que as condições as quais as imagens são coletadas podem variar. Neste caso, normalizar fatores como, por exemplo o Hue, contribuem significativamente para o desempenho do modelo. Por fim, cita-se que em implementações futuras, a expansão do *dataset* seria crucial para melhorar a precisão.

No geral, acredita-se que o protótipo desenvolvido pode ser uma alternativa viável para otimizar e auxiliar os profissionais da Defesa Civil do município de Blumenau e principalmente dos demais municípios do Médio Vale do Itajaí e região. A Defesa Civil de Blumenau possui um catálogo vasto dos deslizamentos ocorridos o qual foi construído ao longo dos anos, isso só foi possível devido a existência de recursos humanos a disposição para realizar a catalogação. Contudo, a grande maioria dos demais municípios do Médio Vale do Itajaí não possuem esses recursos disponíveis. Portanto, a criação dessa base de dados se torna mais trabalhosa e difícil. Com o uso neste protótipo, seria possível construir um sistema ao qual automatizaria a detecção de deslizamentos ocorridos em certa região a partir de imagens de satélite, facilitando o trabalho dos profissionais. Além disso, seria possível realizar inspeções periódicas para acompanhar deslizamentos pré-ocorridos, aumentando indiretamente a capacidade de prevenção de desastres.

4.1 EXTENSÕES

A seguir são listadas possibilidades de extensão deste trabalho:

a) aumentar a quantidade de dados coletados por imagens de sensoriamento remoto, principalmente em relação a outros municípios da região de Blumenau;

b) realizar testes do modelo utilizando vídeos;
c) realizar testes do modelo com imagens disponibilizadas por veículos aéreos não tripulados;
d) testar novos tipos de *data augmentation* e *preprocessing*, visando melhorar a detecção;
e) automatizar a inferência dos deslizamentos de maneira periódica.

REFERÊNCIAS

AUGUSTO FILHO, Oswaldo. **Caracterização geológico-geotécnica voltada à estabilização de encostas:** Uma Proposta Metodológica. In Conferencia Brasileira sobre Estabilidade de Encostas, Rio de Janeiro. Anais. Rio de Janeiro: ABMS/ABGE/PUCRJ, 1992, p.721-733.

BESSA, Gabriela. **Análise da suscetibilidade de ocorrência de quedas de blocos e escorregamentos de solo no Morro da Mariquinha, SC**. Florianópolis, 2015. 135 p.

BIGARELLA, João José. **Estrutura e Origem das Paisagens Tropicais e Subtropicais 3**. 2ª. ed. Florianópolis: Universidade Federal de Santa Catarina, 2007. 552 p

BOCHKOVSKIY, Alexey; WANG, Chien-Yao; LIAO, Hong-Yuan Mark. YOLOv4: Optimal Speed and Accuracy of Object Detection. **ArXiv**, [S.l.], abs/2004.10934, 2020.

BRASIL. CELSO SANTOS CARVALHO. (Org.). **Mapeamento de Risco em Encostas e em Margem de Rios.** Brasília: Ministério das Cidades/instituto de Pesquisas Tecnológicas, 2007. 176 p.

CARVALHO, Celso Santos. e GALVAO, Thiago. **Prevenção de Riscos de Deslizamentos em Encostas: Guia para Elaboração de Políticas Municipais.** Brasília, 2006. Disponível em http://planodiretor.mprs.mp.br/arquivos/prevencaoriscos.pdf. Acesso em 05 dez. 2022.

CHENG, Libo; LI, Jia; DUAN, Ping; WANG, Mingguo. A small attentional YOLO model for landslide detection from satellite remote sensing images. **Landslides**, [S.l.], v. 18, n. 8, p. 2751-2765, 22 maio 2021. Springer Science and Business Media LLC.

COPPOLA, D. P. **Introduction to International Disaster Management.** New York: BH/Elsevier, 2011.

FERNANDES, Jaline Silva de Araujo. Desastres socioambientais: impactos na política de saúde do brasil. **O Social em Questão**, Rio de Janeiro, v. 23, n. 48, p. 243-266, set. 2020.

FERNANDES, Nelson Ferreira. e AMARAL, Cláudio Palmeiro. Movimentos de massa: uma abordagem geológica-geomorfológica. In: GUERRA, Antonio José Teixeira e CUNHA, Sandra Baptista da. **Geomorfologia e Meio Ambiente**. Rio de Janeiro: Bertrand Brasil, 1996, p. 123-186.

FU, Rao *et al*. Fast Seismic Landslide Detection Based on Improved Mask R-CNN. **Remote Sensing**, [S.l.], v. 14, n. 16, p. 3928, 12 ago. 2022.

GHORBANZADEH, Omid; GHOLAMNIA, Khalil; GHAMISI, Pedram. The application of ResU-net and OBIA for landslide detection from multi-temporal sentinel-2 images. **Big Earth Data**, [S.l.], p. 1-26, 14 fev. 2022. Informa UK Limited.

GHORBANZADEH, Omid *et al*. Landslide detection using deep learning and object-based image analysis. **Landslides**, [S.l.], v. 19, n. 4, p. 929-939, 20 jan. 2022. Springer Science and Business Media LLC.

GitHub, ultralytics. Disponível em: https://github.com/ultralytics/ultralytics. Acesso em 03 maio. 2023.

GODOY, Luiz Carlos. **Processos de Dinâmica Superficial**. Ponta Grossa, 2005. (Apostila)

GOUTTE, Cyril; GAUSSIER, Eric. **A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation.** Lecture Notes In Computer Science, [S.l.], p. 345-359, 2005. Springer Berlin Heidelberg

GUIDICINI, Guido; NIEBLE, Carlos. **Estabilidade de Taludes Naturais e de Escavação**. São Paulo: Universidade de São Paulo, 1976. 170 p

GÜNDÜZ, Mehmet Şirin; IşđK, Gültekin. A new YOLO-based method for social distancing from real-time videos. **Neural Computing And Applications**, [S.l.], v. 35, n. 21, p. 15261-15271, 7 abr. 2023. Springer Science and Business Media LLC.

HIGHLAND, Lynn; BOBROWSKY, Peter. Serviço Geológico dos Estados Unidos (Org.). **O Manual do Deslizamento:** Um Guia para a Compreensão do Deslizamento. Reston, Virgínia: USGS (U.S. Geological Survey), 2008. 156 p.

HOLETZ, Mauricio Walter; **Análise para a mitigação e reestruturação das áreas de risco no bairro Garcia (Blumenau - SC):** uma contribuição para a Defesa Civil. 2007. 143 f. Dissertação (Mestrado em Ciência e Tecnologia Ambiental) - Universidade do Vale do Itajaí, Itajaí.

IPCC. **Managing the Risks of Extreme Events and Disasters to Advance Climate Change Adaptation.** Cambridge: Cambridge University Press, 2012.

JONGMANS, Denis *et al*. Geophysical investigation of a large landslide in glaciolacustrine clays in the Trièves area (French Alps). **Engineering Geology**, [S.l.], Elsevier BV. v. 109, n. 1-2, p. 45-56, out. 2009.

KIVRAK, Oğuzhan; GÜRBÜZ, Mustafa Zahid. Performance Comparison of YOLOv3, YOLOv4 and YOLOv5 Algorithms: A Case Study for Poultry Recognition. **European Journal of Science and Technology**, [S.l.], v. 38, p392-397, 26 jul. 2022.

KORMANN, Tanice Cristina; ROBAINA, Luís Eduardo de Souza. Parâmetros geomorfométricos para análise da suscetibilidade a movimentos de massa na área urbana de Blumenau, Santa Catarina. **Geografia Ensino & Pesquisa**, [S.l.], v. 23, p. 42, 10 fev. 2020. Universidad Federal de Santa Maria.

Li, Yao. Landslide Detection of Hyperspectral Remote Sensing Data Based on Deep Learning With Constrains. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, [S.l.], v. 12, abr 2021.

LIN, Tsung-Yi *et al*. Microsoft COCO: common objects in context. **Computer Vision – Eccv 2014**, [S.l.], p. 740-755, 2014. Springer International Publishing.

MATTEDI, M. A. **As enchentes como tragédias anunciadas: impactos da problemática ambiental nas situações de emergência em Santa Catarina.** Campinas: (Tese de Doutorado – UNICAMP), 1999.

MATTEDI, M. A. **Dilemas e perspectivas da abordagem sociológica dos desastres.** Tempo Social, V. 29, n. 3, pp. 261-285, 2017.

MORA, Camilo *et al.* Broad threat to humanity from cumulative climate hazards intensified by greenhouse gas emissions. **Nature Climate Change**, [S.l.], n. 8, pp. 1062-1071, 2018.

Prefeitura de Blumenau. Dados Geográficos. Disponível em: https://www.blumenau.sc.gov.br/blumenau/as5d1a5sd4a4sd. Acesso em 10 abr. 2023.

REDMON Joseph; FARHADI, Ali. **YOLOv3**: An Incremental Improvement. Washington: **ArXiv**, [S.l.], 2018.

Tan, Mingxing, *et al.* EfficientDet: Scalable and Efficient Object Detection. **2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**, IEEE, 2020, pp. 10778–87.

TIERNEY, K. **Disasters: a sociological approch.** London: Polity Press, 2020.

TOBIN, G. A; MONTZ, B. E. **Natural hazards**: explanation and integration. London: The Guilford Press, 1997.

TOMINAGA, Lídia. **Avaliação de Metodologias de Análise de Risco a Escorregamentos**:Aplicação de um ensaio em Ubatuba, SP. 2007. 220 f. Dissertação (Mestrado) - Curso de Geografia Física, Geografia, USP, São Paulo, 2007.

ULLO, Silvia *et al*. A New Mask R-CNN-Based Method for Improved Landslide Detection. **Ieee Journal Of Selected Topics In Applied Earth Observations And Remote Sensing**, [S.l.], v. 14, p. 3799-3810, 2021.

Ultralytics, YOLOv8 Documentation. Disponível em: https://docs.ultralytics.com/. Acesso em 03 maio. 2023.

UNITED STATES GEOLOGICAL SURVEY. **Landslide Types and Processes.** 2004. Disponível em: https://pubs.usgs.gov/fs/2004/3072/fs-2004-3072.html. Acesso em 20 set. 2022.

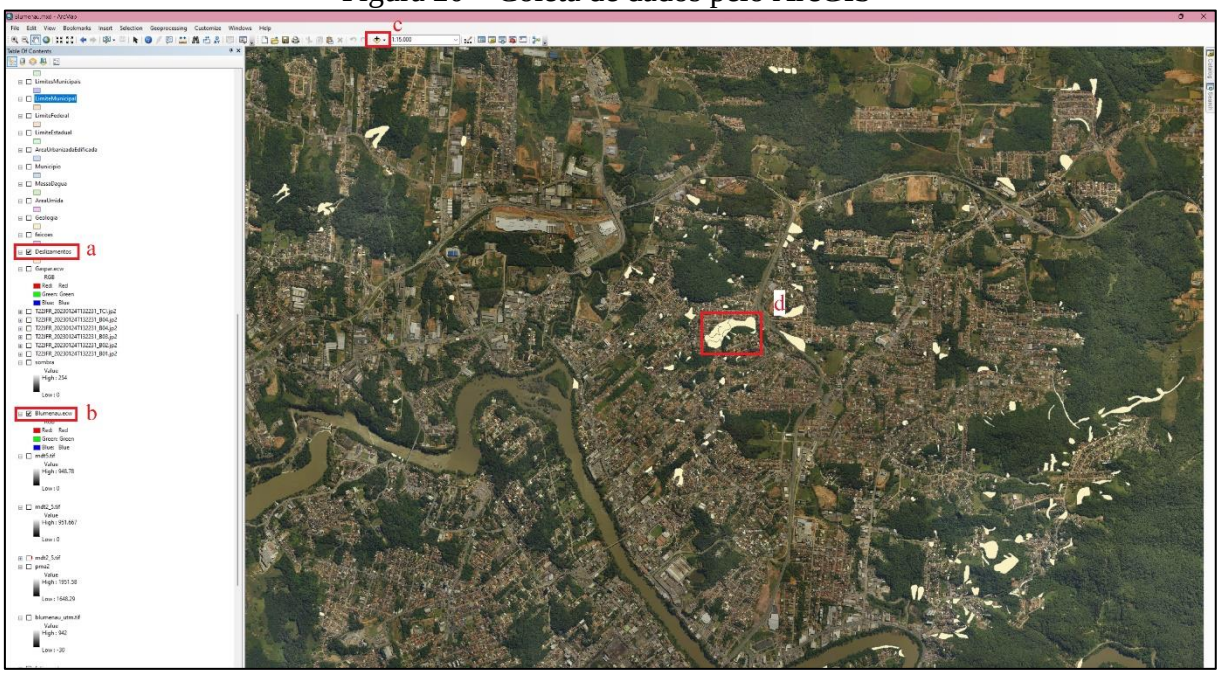
YANG, Ruilin et al. **Landslide Extraction Using Mask R-CNN with Background-Enhancement Method**. Remote Sensing, [S.l.], v. 14, n. 9, p. 2206, 5 maio 2022.

Yi, Yaning, and Wanchang Zhang. A New Deep-Learning-Based Approach for Earthquake-Triggered Landslide Detection From Single-Temporal RapidEye Satellite Imagery. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, [S.l.], vol. 13, 2020, pp. 6166–76.

APÊNDICE A – Coleta e preparação das imagens

Durante a coleta das imagens, foram utilizados os softwares Google Earth Pro e ArcGIS. A Figura 20 demonstra o processo de coleta de dados pelo software ArcGIS, onde no item “a” encontra-se a camada disponibilizada pela Defesa Civil com as demarcações de deslizamentos catalogados manualmente. Este dado foi utilizado como base para encontrar os objetos utilizados para a criação do *dataset*. O item “b” contém a camada com o mapa do município de Blumenau, a qual foi disponibilizada publicamente pelo CPRM. O item “c” apresenta o botão onde é possível realizar a importação dos dados. Por fim, o item “d” demonstra um exemplo de objeto catalogado pela Defesa Civil.

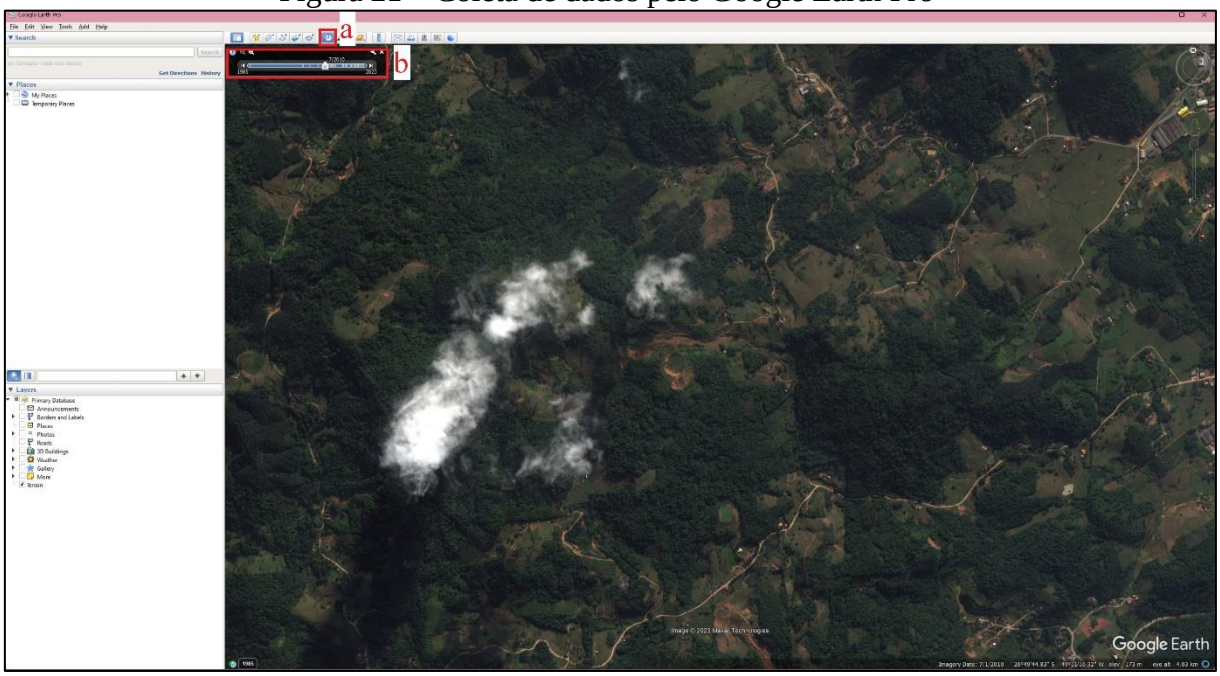
Figura 20 – Coleta de dados pelo ArcGIS



Fonte: elaborado pelo autor.

A coleta de dados no Google Earth Pro foi feita selecionando o menu existente no item “a”, escolhendo a data no item “b”. A localização dos objetos deve ser feita manualmente a fim de correlacionar a posição geográfica no Google Earth Pro com a disponibilizada pela Defesa Civíl no ArcGIS, conforme ilustra a Figura 21.

Figura 21 – Coleta de dados pelo Google Earth Pro

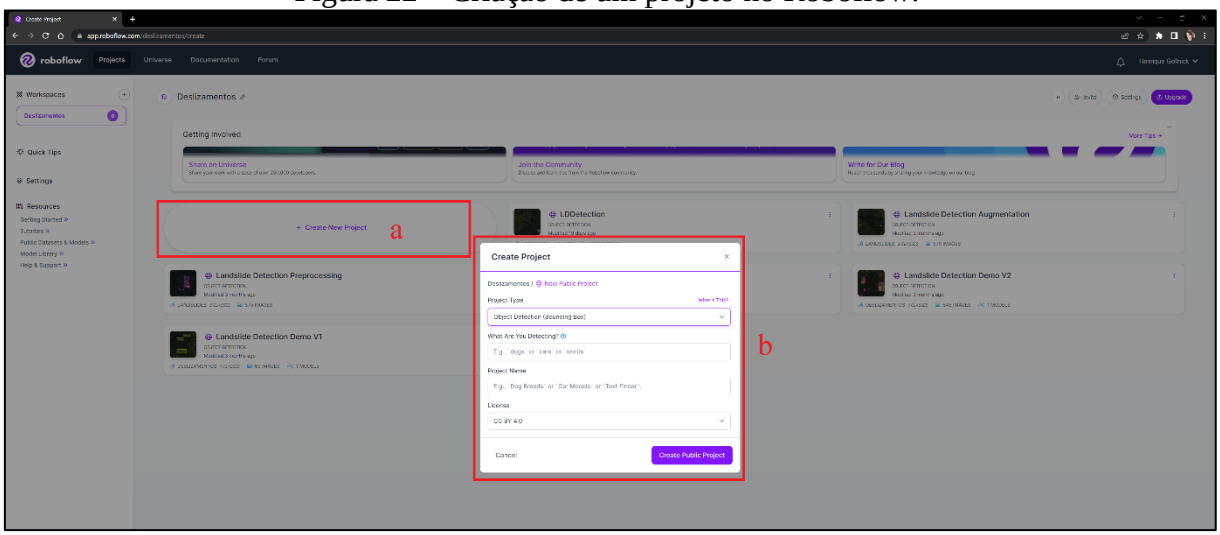


Fonte: elaborado pelo autor.

Apêndice B – Ambiente Roboflow

Para preparar e versionar o *dataset* utilizou-se o site Roboflow. Inicialmente é necessário criar um projeto, isto é possível através do botão do item “a” da Figura 22, ao qual solicitará o preenchendo os dados na caixa de diálogo demonstrado no item “b” da Figura 22.

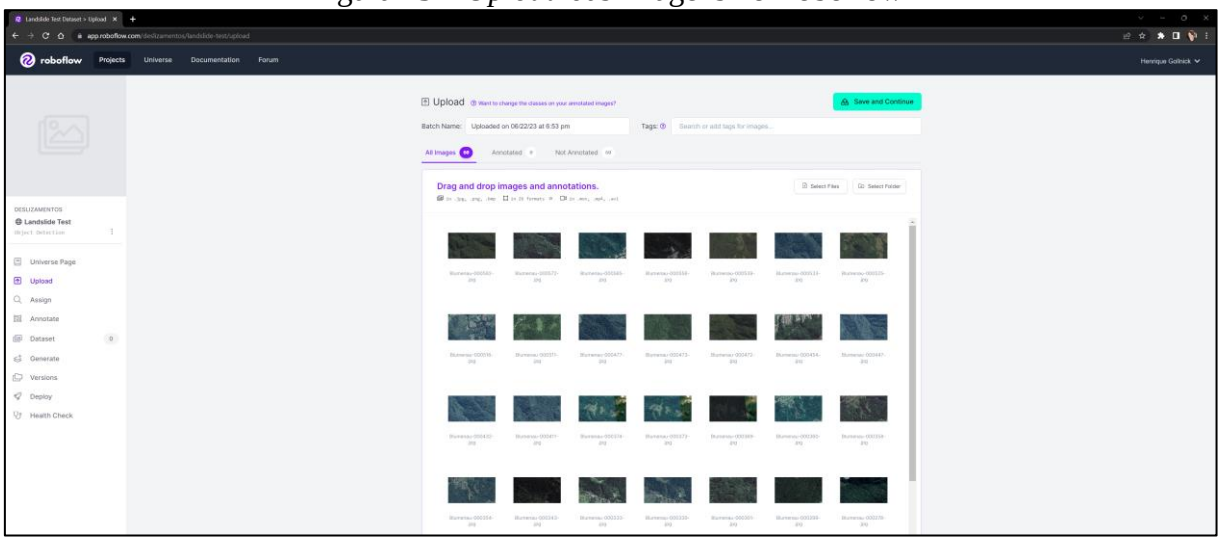
Figura 22 – Criação de um projeto no Roboflow.



Fonte: elaborado pelo autor.

A Figura 23 apresenta a tela principal após a criação do projeto com os dados solicitados. Nela é possível realizar o *upload* das imagens que irão compor o *dataset* e que posteriormente serão anotadas/demarcadas. O usuário pode arrastas as imagens para a tela, sendo automaticamente importadas. Ressalta-se que os formatos suportados são: .jpg, .png e .bmp; e para vídeos são: .mov, .mp4 e .avi.

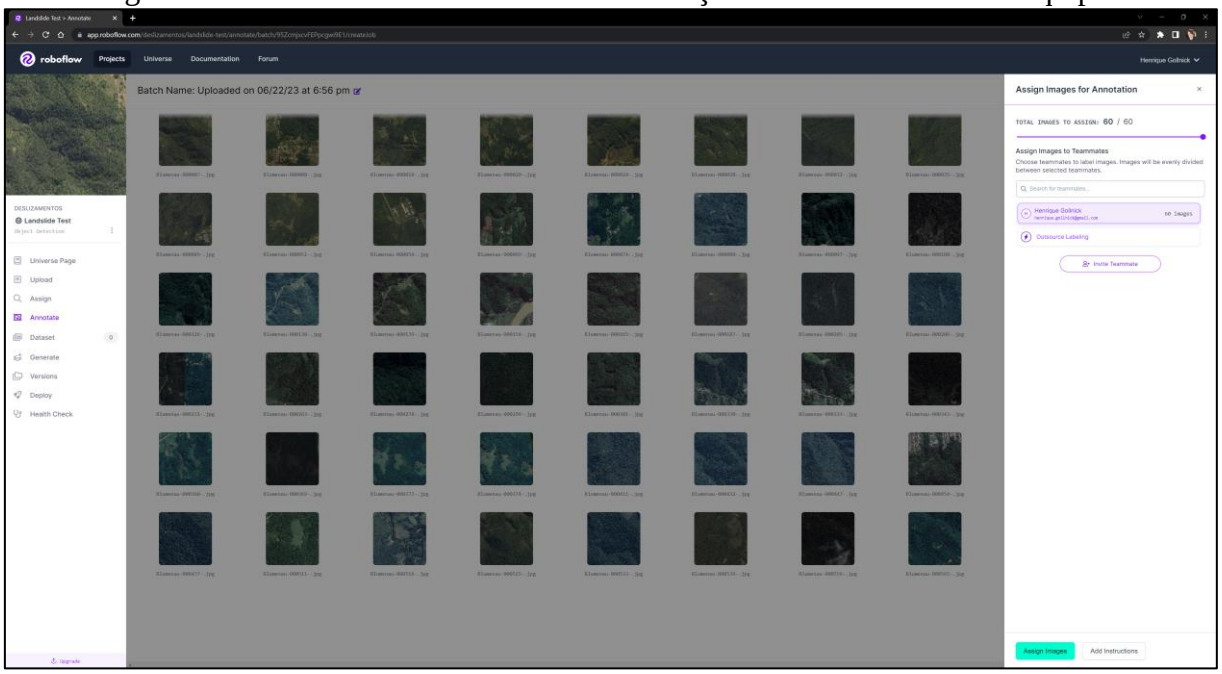
Figura 23 – *Upload* das imagens no Roboflow



Fonte: elaborado pelo autor.

Caso o usuário queira trabalhar em equipe, é possível adicionar instruções de demarcação. Deste modo é possível adicionar instruções sobre como devem ser realizadas as anotações nas imagens. A Figura 24 exibe a tela de anotação em equipe do site Roboflow.

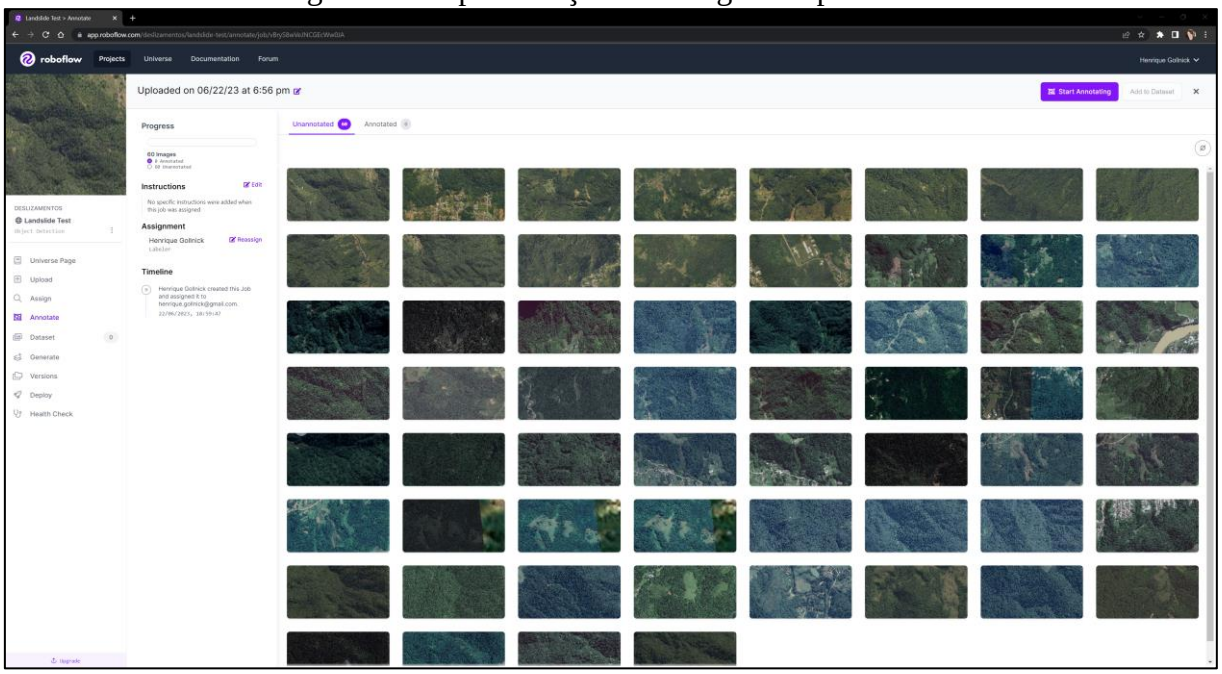
Figura 24 – Divisão das atividades de demarcação entre os membros da equipe.



Fonte: elaborado pelo autor.

Após o *upload*, as imagens são listadas para o usuário, conforme mostra a Figura 25.

Figura 25 – Apresentação das imagens importadas.

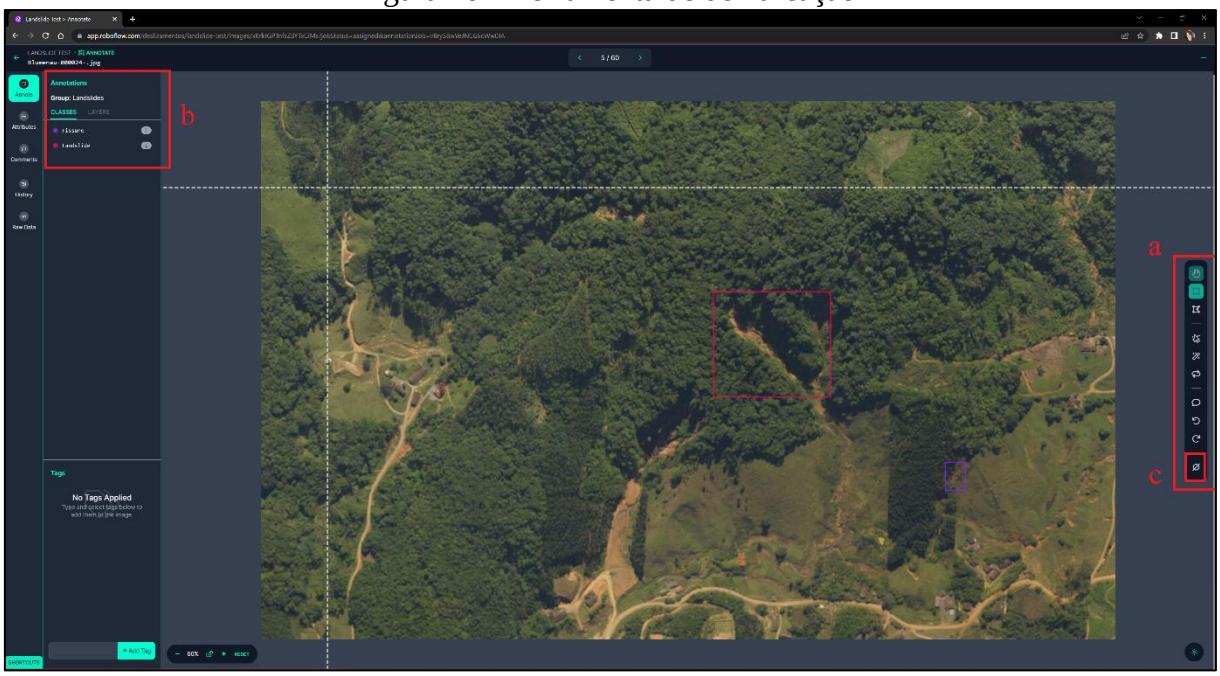


Fonte: elaborado pelo autor.

A demarcação pode ser feita utilizando a *toolbar* exemplificada pelo item “a” da Figura 26, juntamente com a criação de *labels* realizado pelo menu de contexto apresentado pelo item

“b” da Figura 26. Caso não existam objetos na imagem, é possível demarcá-la como nula através do botão apresentado pelo item “c” da Figura 26.

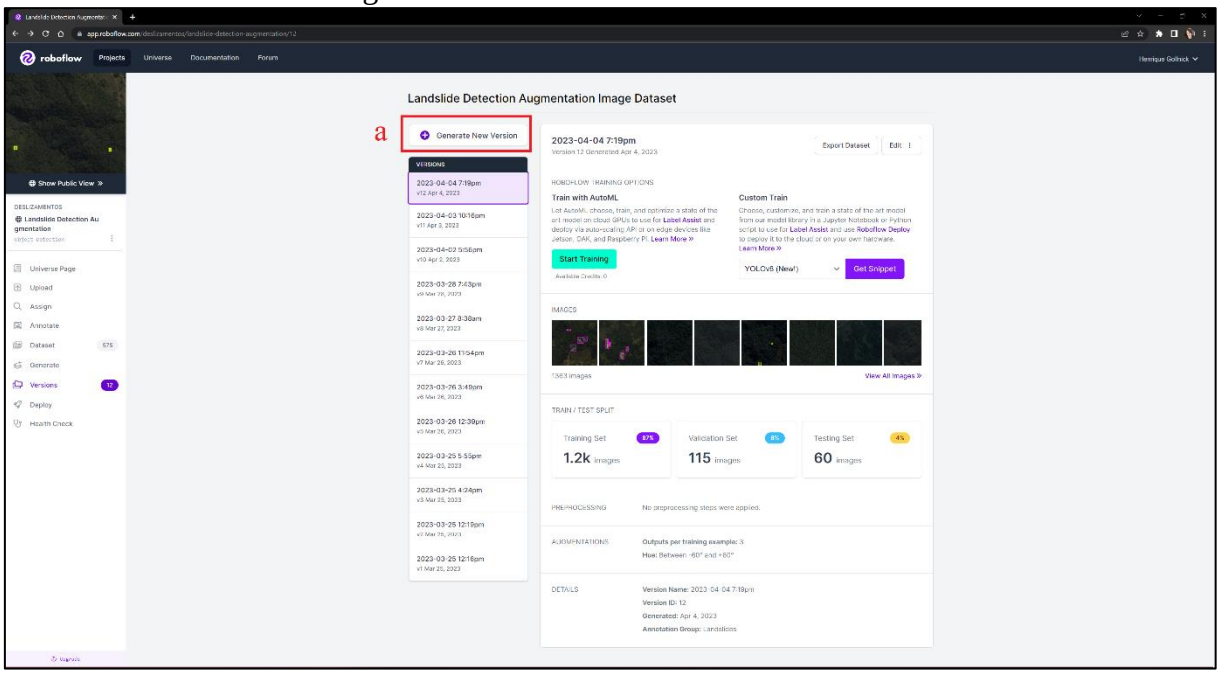
Figura 26 – Ferramenta de demarcação



Fonte: elaborado pelo autor.

Posteriormente, é possível gerenciar o projeto a partir de seu *dashboard*. Novas versões do *dataset* podem ser criadas pelo botão exemplificado pelo item “a” da Figura 27.

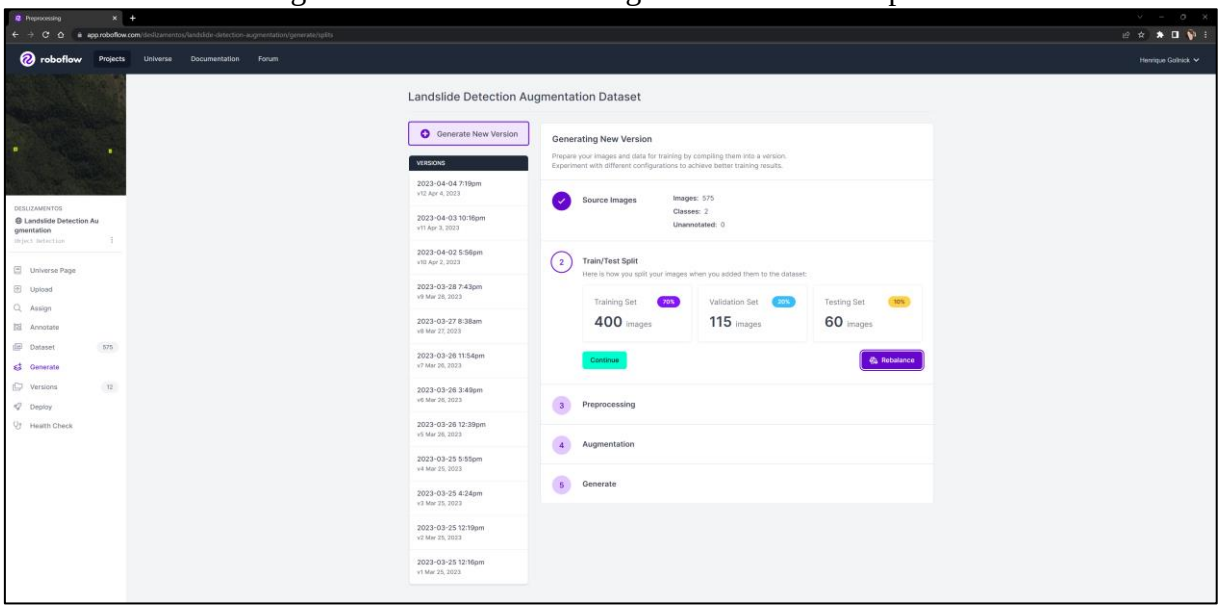
Figura 27 – *Dashboard* de versionamento.



Fonte: elaborado pelo autor.

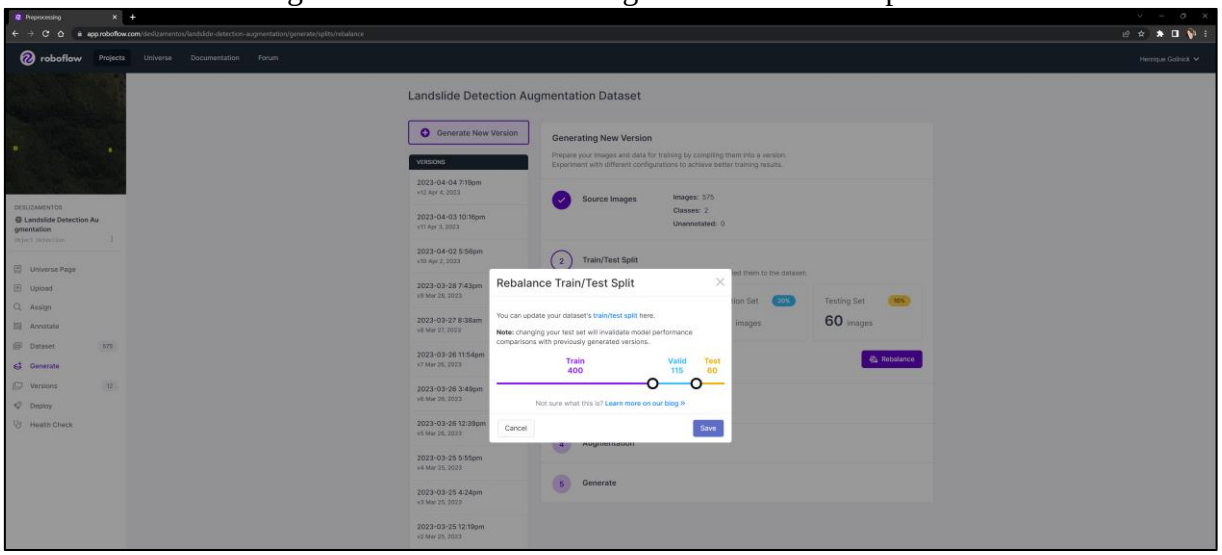
Ao gerar uma nova versão do *dataset*, é necessário selecionar a divisão do número de imagens de treino, validação e teste. A Figura 28 demonstra o menu de *split* presente na tela de versionamento e a Figura 29 apresenta o menu no qual é possível aplicar a técnica.

Figura 28 – Divisão das imagens do *dataset* - etapa A



Fonte: elaborado pelo autor.

Figura 29 – Divisão das imagens do *dataset* - etapa B.

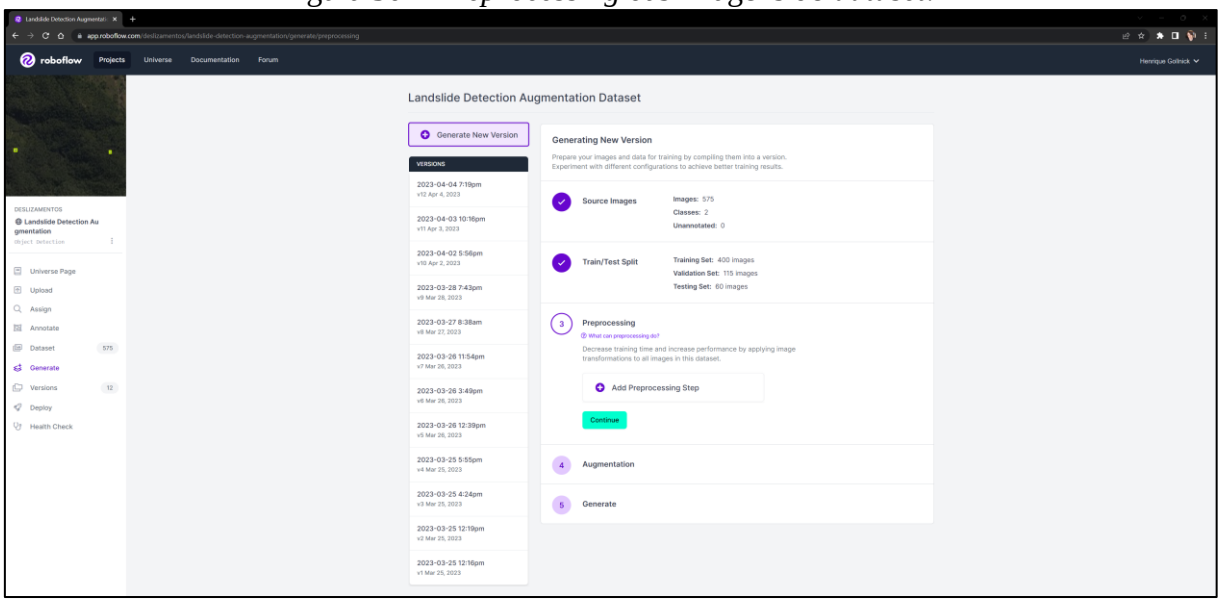


Fonte: elaborado pelo autor.

A partir disso, pode-se selecionar as etapas de *preprocessing*. Esta etapa normaliza os dados presentes no *dataset*. Os possíveis *preprocessing* são: *auto-orient, isolate objects, static crop, resiz, grayscale, auto-adjust contrast, tile, modify classes e filter null*. Não é necessário que o usuário aplique quaisquer etapas, contudo, é altamente recomendado que alguns *preprocessings* sejam exercidos na maioria dos *datasets*, como é o caso do *auto-orient.* Cada tipo de *preprocessing* possui suas peculiaridades, como por exemplo o *tile*, em que é necessário escolher a dimensão a qual as *tiles* serão criadas. A Figura 30 apresenta a tela inicial de

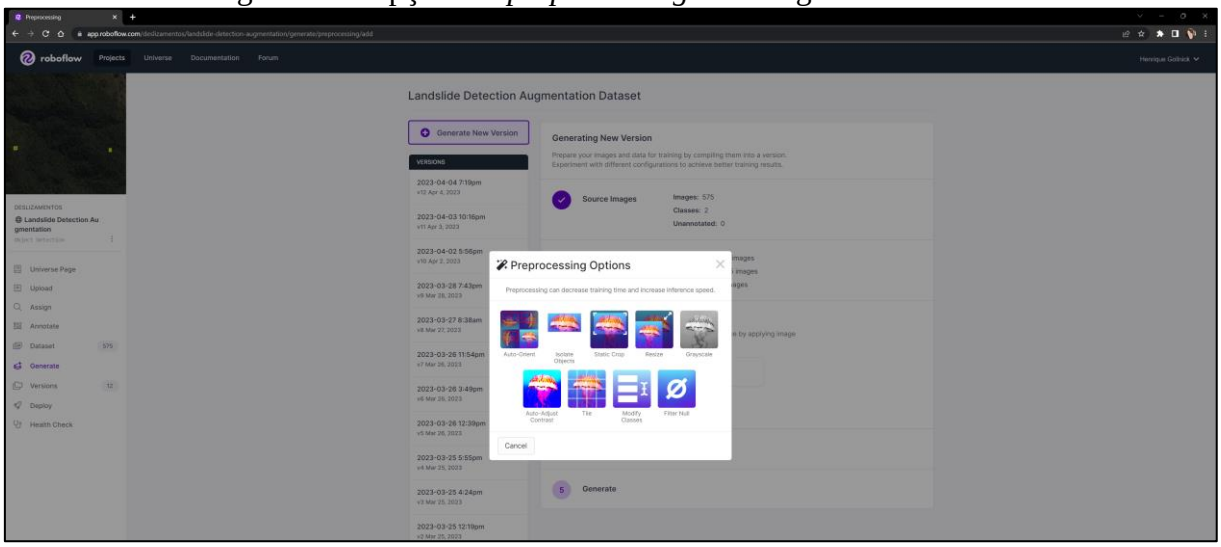
selecionar possíveis passos de *preprocessing*, e a Figura 31 demonstra o diálogo onde é possível escolher a técnica de *preprocessing* desejada.

Figura 30 – *Preprocessing* das imagens do *dataset*.



Fonte: elaborado pelo autor.

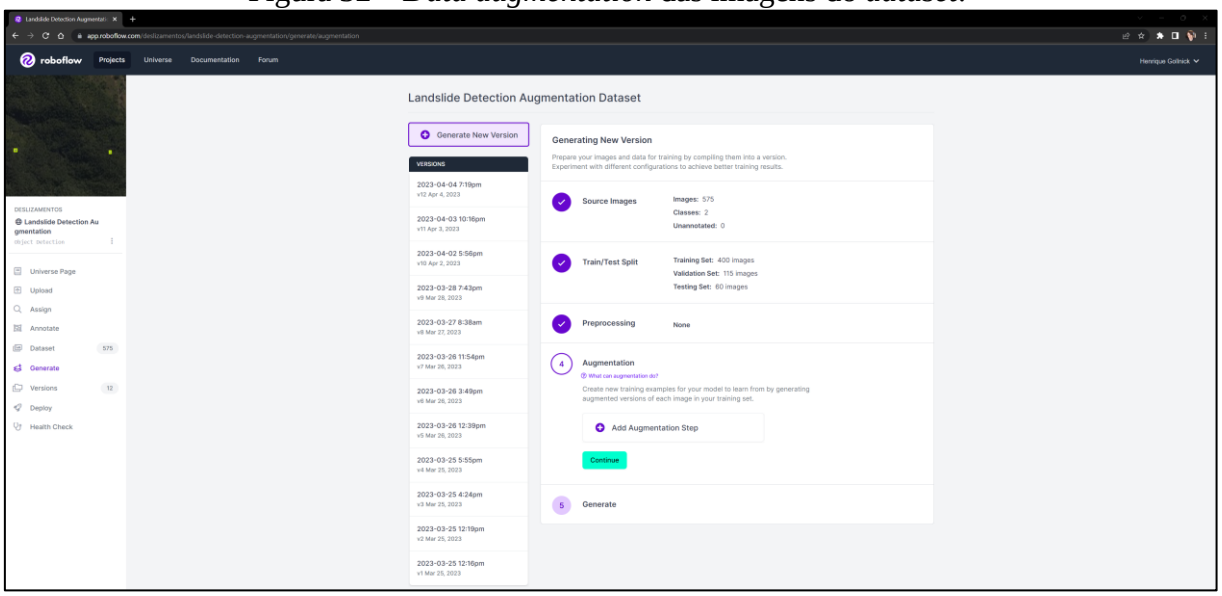
Figura 31 – Opções de *preprocessing* das imagens do *dataset*.



Fonte: elaborado pelo autor.

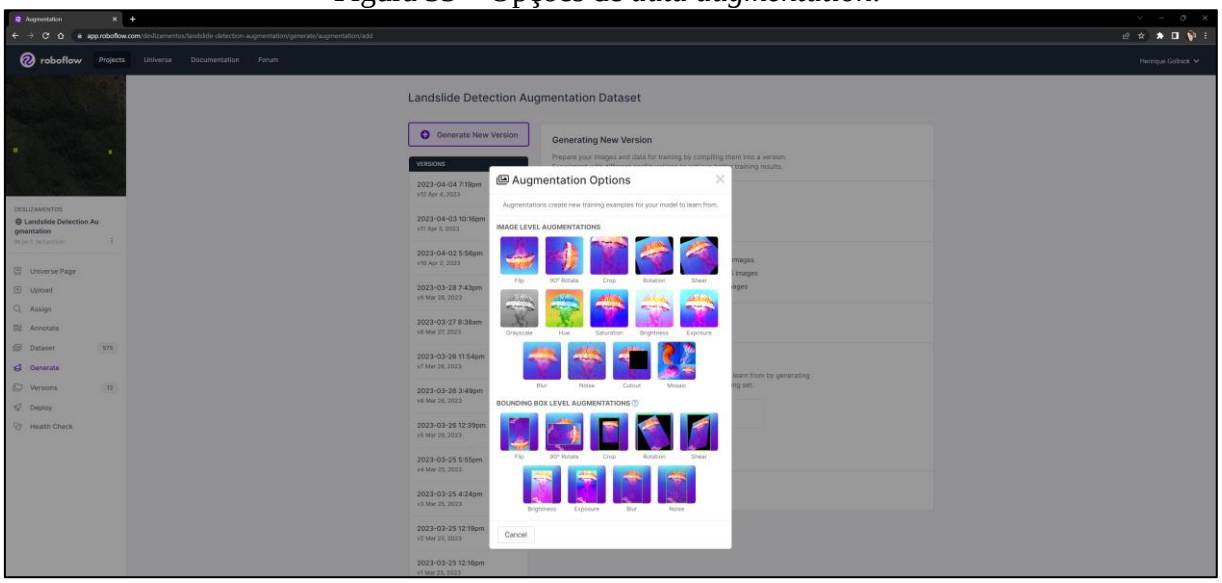
Posteriormente é realizada a etapa de *data augmentation*. Esta etapa aumentara a variedade das imagens presentes no *dataset*. A Figura 32 apresenta a tela inicial onde se pode selecionar possíveis passos de *data augmentation*, e a Figura 33 demonstra o diálogo no qual possível escolher a técnica de *data augmentation* desejada.

Figura 32 – *Data augmentation* das imagens do *dataset*.



Fonte: elaborado pelo autor.

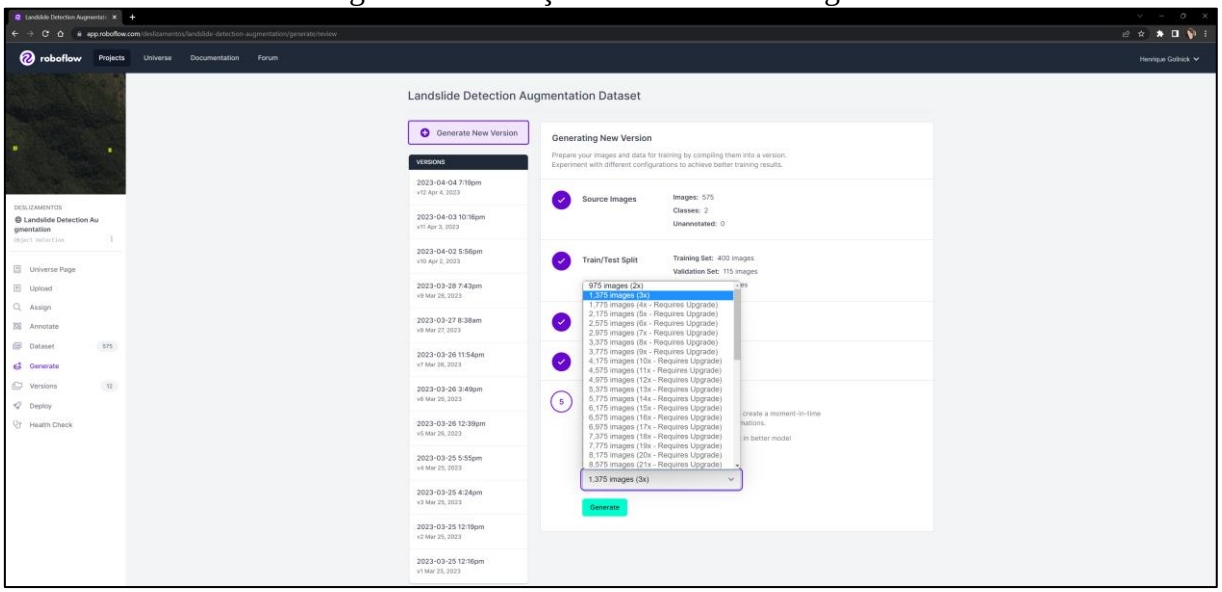
Figura 33 – Opções de *data augmentation*.



Fonte: elaborado pelo autor.

Por fim, é possível selecionar o número de imagens a serem geradas. Contudo, o número de imagens disponíveis dependerá dos tipos de *preprocessing* e *data augmentation* selecionados. A Figura 34 demonstra como escolher o número de imagens que irão compor o *dataset*.

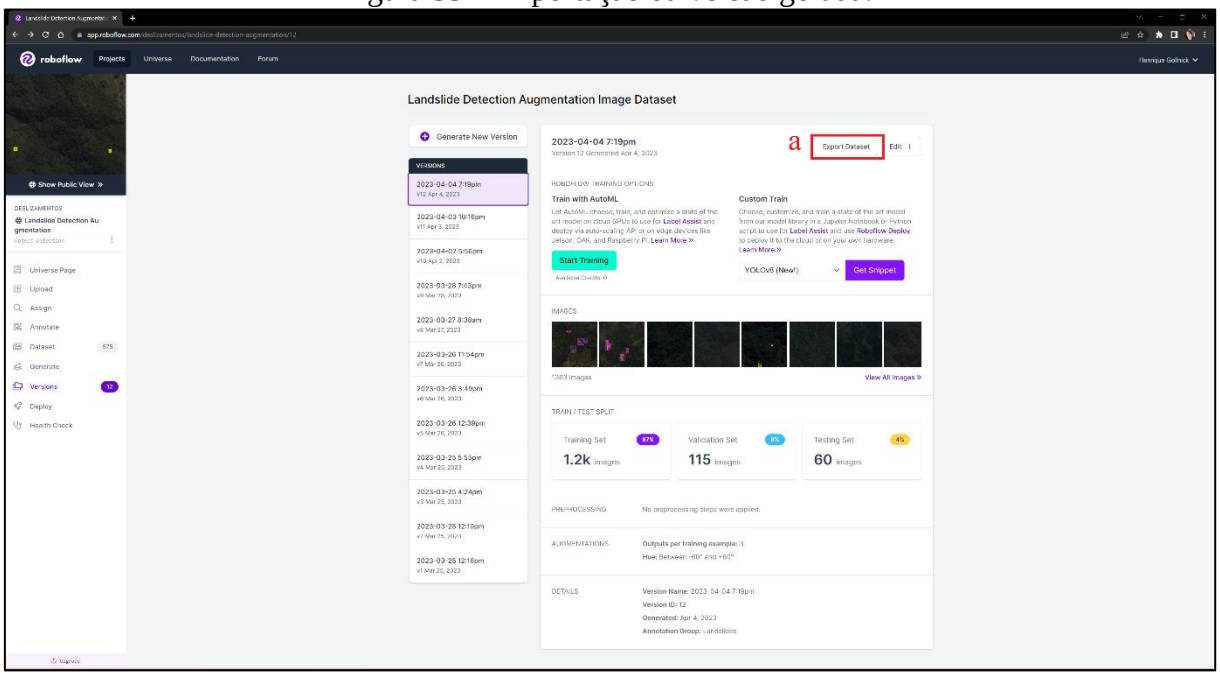
Figura 34 – Geração do número de imagens.



Fonte: elaborado pelo autor.

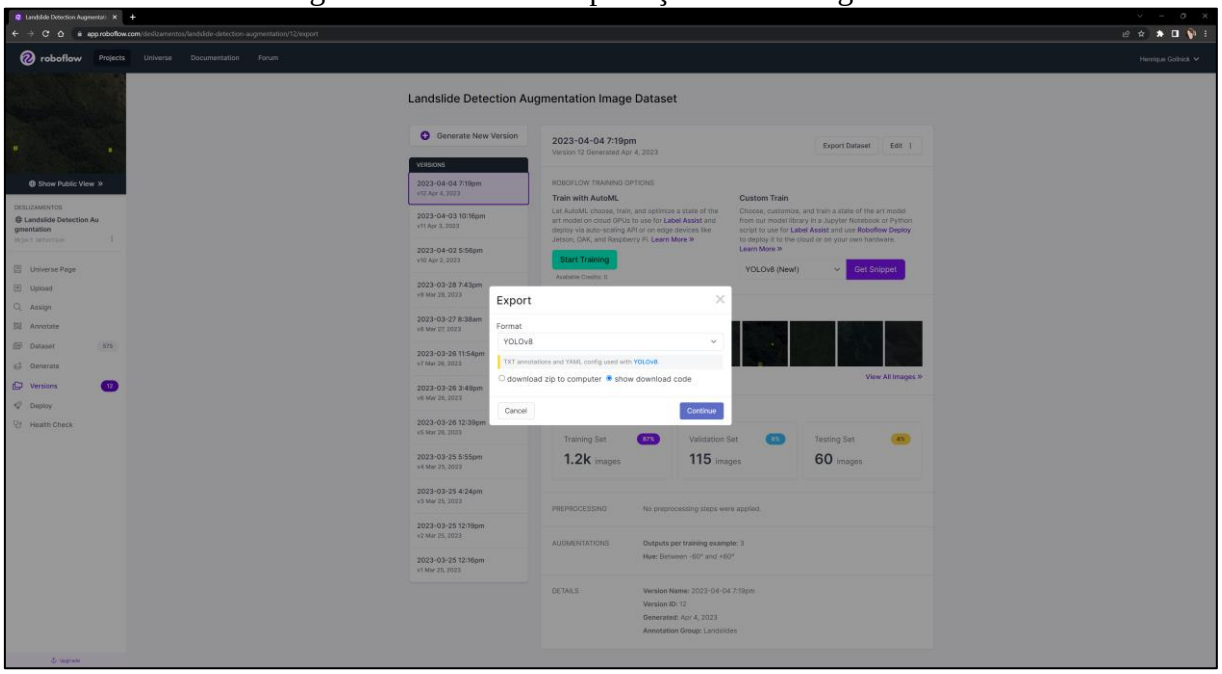
A Figura 35 exemplifica como exportar a versão no *dashboard* (item “a” da Figura 35). Vale ressaltar que é possível gerar um código fonte, comando ou url para baixar o *dataset* diretamente ou exportar como arquivo zip. O Modelo escolhido para as anotações também deve ser selecionado. Neste caso, o modelo selecionado foi YOLOv8. A Figura 36 apresenta a caixa de diálogo utilizara para selecionar o modelo da versão, e a Figura 37 demonstra o código gerado pelo *export*.

Figura 35 – Exportação da versão gerada.



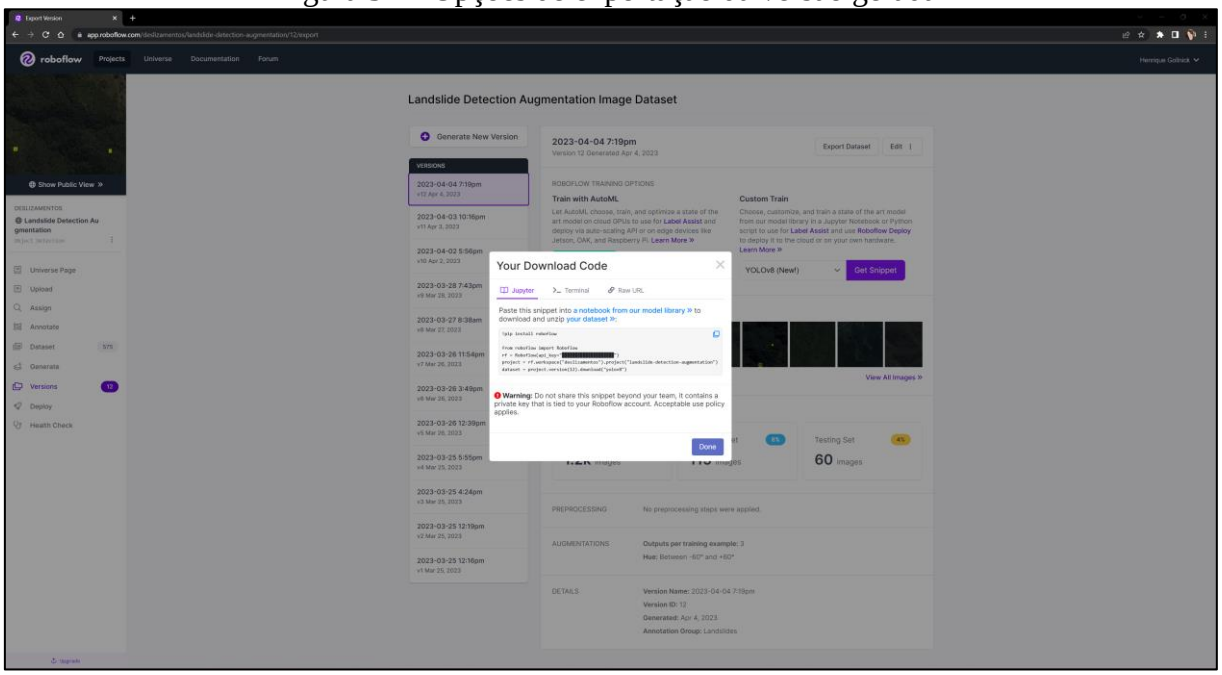
Fonte: elaborado pelo autor.

Figura 36 – Menu de exportação da versão gerada.



Fonte: elaborado pelo autor.

Figura 37 – Opções de exportação da versão gerada



Fonte: elaborado pelo autor.