Mapeamento de Formigueiros em Ortofotos Aéreas por Segmentação Semântica com U-Net

Isabelle Oliveira Bicudo[1], Rodolpho Ale[1]

Orientador: Prof. Dr. Wesley Nunes Gonçalves

[1]Faculdade de Computação Universidade Federal de Mato Grosso do Sul (UFMS)
Campo Grande MS Brasil

Resumo

Este trabalho apresenta uma abordagem de segmentação semântica para detecção automática de formigueiros em ortofotos aéreas de talhões agrícolas. O problema é desafiador devido ao forte desbalanceamento entre fundo e classe de interesse, à baixa frequência de exemplos positivos e à semelhança visual entre formigueiros e regiões de solo exposto. A solução proposta utiliza uma arquitetura U-Net para segmentação binária, com pré-processamento das máscaras, normalização das imagens, transformações sincronizadas entre imagem e rótulo, aumento de dados da classe de interesse e pós-processamento por limiar de confiança e filtragem de regiões. Foram avaliadas diferentes configurações de treinamento e validação, considerando métricas de detecção e segmentação, como Precisão, Revocação, F1, Dice e IoU. Os resultados indicam que a preparação dos dados e o tratamento do desbalanceamento, em conjunto com a estabilização da arquitetura (normalização e superamostragem aprendida), foram determinantes para a evolução do modelo. A melhor configuração priorizou a detecção de formigueiros, alcançando Revocação de 91,7%, F1 de 83,1% e IoU da classe formigueiro de 33,8%. A análise dos erros mostrou que falsos positivos estão associados principalmente a solo avermelhado e marcas no terreno, enquanto falsos negativos ocorrem em formigueiros pequenos ou visualmente semelhantes ao fundo.

Palavras-chave: segmentação semântica; U-Net; formigueiros; ortofotos aéreas; visão computacional agrícola; desbalanceamento de classes.

1 Introdução

O monitoramento de pragas em lavouras é uma atividade essencial para reduzir perdas produtivas e orientar ações de manejo. Entre os problemas recorrentes em áreas agrícolas, a presença de formigueiros pode comprometer o desenvolvimento da cultura e exigir intervenções localizadas [13]. Em grandes talhões, a inspeção manual demanda tempo, custo operacional e depende da experiência dos avaliadores em campo.

O uso de veículos aéreos não tripulados e ortofotos de alta resolução permite observar grandes extensões agrícolas com maior frequência [14]. Entretanto, a análise manual dessas imagens continua sendo trabalhosa, especialmente quando o alvo possui baixa representatividade espacial. Nesse contexto, técnicas de visão computacional e aprendizado profundo podem auxiliar na identificação automática de regiões de interesse [9].

Este trabalho aborda a detecção de formigueiros como um problema de segmentação semântica binária. Dado um recorte RGB de uma ortofoto aérea, o objetivo é classificar cada pixel como fundo ou formigueiro. A segmentação semântica é adequada ao problema porque os formigueiros possuem formato irregular, o que torna a representação por caixas delimitadoras menos precisa.

Os principais desafios encontrados foram: (i) predominância de pixels de fundo; (ii) pequena área ocupada pelos formigueiros, mesmo nas imagens positivas; e (iii) semelhança visual entre solo exposto e formigueiros. Para enfrentar esses fatores, foi desenvolvido um fluxo de processamento (*pipeline*) baseado em U-Net, com pré-processamento das máscaras, normalização das imagens, transformações sincronizadas entre imagem e máscara, aumento de dados (*data augmentation*) da classe minoritária e análise sistemática das métricas de validação. A hipótese é que, em cenários de segmentação semântica com forte desbalanceamento e poucos exemplos positivos, estratégias sistemáticas de preparação de dados e de tratamento do desbalanceamento (padronização de máscaras, transformações sincronizadas, aumento de dados da classe minoritária, funções de custo especializadas e calibração de limiar) produzem ganhos substanciais e incrementais na detecção e na segmentação da classe minoritária.

As contribuições deste trabalho são:

- avaliação sistemática do impacto da preparação de dados em segmentação semântica altamente desbalanceada;
- análise experimental do compromisso (*trade-off*) entre Revocação (*recall*) e IoU (interseção sobre união) em detecção de formigueiros;
- proposta de fluxo de processamento robusto para tratamento de máscaras RGB em segmentação agrícola;
- investigação do impacto combinado das perdas de Tversky (*Tversky Loss*), Focal (*Focal Loss*) e Lovász (*Lovász Loss*) em cenários de baixa representatividade da classe positiva;
- análise qualitativa dos principais padrões de erro do modelo.

2 Trabalhos Relacionados

A segmentação semântica com redes neurais convolucionais foi consolidada a partir das Fully Convolutional Networks (FCN) [1], que substituíram camadas densas por convoluções para gerar mapas densos de classificação. Posteriormente, a U-Net [2] tornou-se uma das arquiteturas mais utilizadas em segmentação por combinar um caminho de contração, responsável por capturar contexto, com um caminho de expansão, responsável por recuperar resolução espacial. As conexões diretas entre codificador (*encoder*) e decodificador (*decoder*) permitem preservar informações finas de localização, sendo úteis para objetos pequenos e irregulares.

Em tarefas agrícolas, redes profundas têm sido aplicadas a imagens de drones e satélites para identificação de plantas daninhas [11, 12], doenças [10], falhas de plantio e degradação do solo [9]. Estudos nessa área indicam que modelos convolucionais são adequados para padrões visuais locais, mas sofrem com variações de iluminação, escala e textura. Problemas de segmentação em imagens aéreas também costumam apresentar desbalanceamento severo, pois a classe de interesse ocupa pequena fração dos pixels [12].

Para lidar com esse cenário, funções de custo baseadas em sobreposição, como a perda Dice (*Dice Loss*) [4] e a perda de Tversky [5], são utilizadas para reduzir o domínio da classe majoritária. A perda Focal [3] também é empregada em problemas com grande quantidade de exemplos fáceis, pois reduz a contribuição de pixels classificados corretamente com alta confiança. Além disso, estratégias de aumento de dados [8] e seleção de recortes (*patches*) [2] são importantes para aumentar a exposição do modelo a regiões relevantes.

Especificamente para a detecção de formigueiros (ninhos de formigas cortadeiras), trabalhos recentes empregaram sensoriamento remoto: Santos et al. [16] mapearam ninhos de *Atta sexdens* em plantios de teca a partir de imagens de satélite, e dos Santos et al. [15] detectaram e mediram ninhos com aprendizado profundo sobre imagens de VANT (Veículo Aéreo Não Tripulado).

Este trabalho diferencia-se por abordar a detecção de formigueiros como segmentação semântica pixel a pixel e por avaliar sistematicamente o impacto da preparação de dados e do tratamento do desbalanceamento no desempenho. Além disso, separa formalmente métricas de detecção e de segmentação, permitindo análise mais precisa do compromisso entre Revocação e IoU em cenários de objetos pequenos e classes altamente desbalanceadas.

3 Materiais e Métodos

A Figura 1 resume o fluxo de processamento proposto. De forma resumida, o processamento ocorre nas seguintes etapas:

1. **Leitura dos dados:** carrega-se o recorte RGB da ortofoto e a máscara correspondente, que marca onde estão os formigueiros.
2. **Decodificação da máscara:** a máscara vem com regiões pintadas em cores; cada cor é convertida em um rótulo numérico (fundo, formigueiro ou área ignorada) que o modelo consegue interpretar.
3. **Normalização e transformações sincronizadas:** os valores dos pixels são ajustados a uma escala comum e transformações geométricas (rotações e espelhamentos) são aplicadas de forma idêntica à imagem e à máscara, para que permaneçam alinhadas.
4. **Predição pela U-Net:** a rede analisa o recorte e atribui a cada pixel uma probabilidade de pertencer a um formigueiro.
5. **Pós-processamento:** um limiar de confiança decide quais pixels são considerados formigueiro e um filtro remove regiões muito pequenas ou improváveis, gerando a máscara final.

3.1 Conjunto de dados

O conjunto de dados (*dataset*) é composto por recortes de ortofotos aéreas RGB e suas respectivas máscaras de segmentação. As imagens utilizadas possuem dimensão padrão de 256×256 pixels. As máscaras foram produzidas manualmente e codificadas por cores, sendo necessário convertê-las para rótulos numéricos antes do treinamento.

Esse conjunto foi disponibilizado já preparado, segmentado em recortes de 256×256 pixels e contendo exclusivamente imagens RGB de um único levantamento aéreo, sem bandas espectrais adicionais (como o infravermelho próximo) nem informação temporal sobre o estágio de desenvolvimento dos formigueiros. O tamanho dos recortes, herdado desse formato, é também compatível com a memória da GPU utilizada (6 GB), que restringe o tamanho de lote a duas imagens. Por terem sido recebidos já recortados e apenas em RGB, não foi possível avaliar três aspectos, deixados como trabalho futuro: (i) se recortes maiores, ao ampliar o contexto ao redor de cada formigueiro, beneficiariam a classe de interesse, dada sua pequena proporção frente ao fundo; (ii) como o estágio (idade) do formigueiro afeta a detecção, ainda que formigueiros maiores tendam a se destacar mais da vegetação; e (iii) o uso de outros espectros para separar melhor vegetação e formigueiros.

A Tabela 1 apresenta a distribuição das imagens utilizadas nos experimentos, indicando quantas contêm formigueiro (positivas) e quantas não contêm (negativas). Os valores referem-se ao número de imagens (recortes de 256×256 px), não a pixels.

Tabela 1: Distribuição do conjunto de dados, em número de imagens (256×256 px) por conjunto.

Conjunto	Imagens	Positivas	Negativas
Treino	9.862	2.406	7.456
Validação	2.466	593	1.873

Embora cerca de 24% das imagens contenham formigueiro (proporção semelhante no treino e na validação), o desbalanceamento do problema manifesta-se sobretudo no nível de pixel: os formigueiros representam menos de 1% do total de pixels do conjunto. Assim, mesmo quando uma imagem contém formigueiro, a área ocupada pela classe de interesse é muito pequena, o que torna a segmentação desafiadora. A Figura 2 ilustra um recorte positivo, sua máscara e um recorte negativo de solo avermelhado.

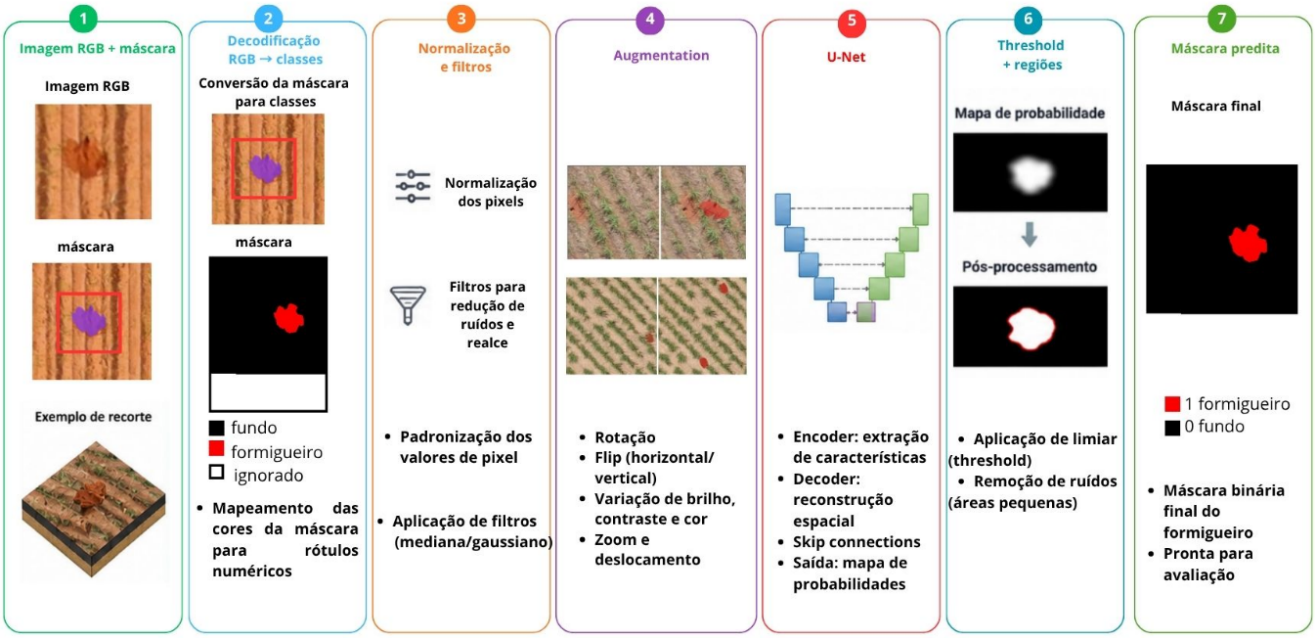
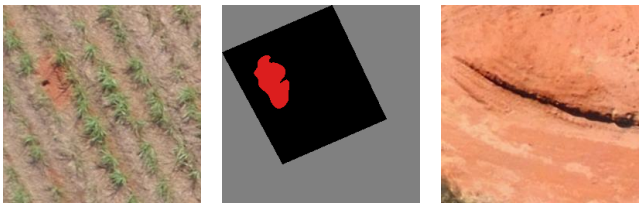


Figura 1: Visão geral do fluxo de processamento de segmentação semântica para detecção de formigueiros.



(a) RGB **(b)** Máscara **(c)** Negativo

Figura 2: Exemplos do conjunto de dados: (a) recorte RGB com formigueiro; (b) máscara correspondente (vermelho: formigueiro; preto: fundo; cinza: área ignorada); (c) recorte negativo de solo avermelhado, visualmente semelhante a um formigueiro.

Pré-processamento das máscaras. As máscaras RGB foram convertidas para três classes:

- fundo: classe 0;
- formigueiro: classe 1;
- ignorado: classe 255.

A classe 255 foi utilizada para regiões não anotadas ou que não deveriam contribuir para o cálculo da função de perda. Esse tratamento evita que o modelo aprenda padrões incorretos em áreas fora da região de interesse.

Transformações e normalização. As imagens foram normalizadas com média e desvio padrão do ImageNet. Embora o modelo tenha sido treinado no domínio agrícola, essa normalização contribui para estabilizar a escala dos dados de entrada. As transformações geométricas, como rotações e espelhamentos, foram aplicadas de forma sincronizada entre imagem e máscara. Essa decisão é crítica em segmentação semântica, pois qualquer desalinhamento entre imagem e rótulo compromete diretamente o aprendizado.

Tratamento do desbalanceamento no nível de dados. O treinamento utiliza diretamente os recortes de 256×256 (cada recorte é um pedaço da ortofoto). Duas medidas reduzem o desbalanceamento ainda na preparação dos dados:

- **Descarte de recortes quase vazios:** recortes cuja máscara tem mais de 70% de área não anotada (classe *ignorar*, típica das bordas) são removidos do treino, pois quase não trazem sinal de aprendizado e desperdiçam processamento.
- **Duplicação de formigueiros:** para o modelo ver mais exemplos da classe de interesse sem precisar de novas imagens, os formigueiros presentes em um recorte são copiados para outras posições do próprio recorte. Isso ocorre em 70% dos recortes com formigueiro, gerando até duas cópias, cada uma girada ou espelhada aleatoriamente; como a cópia permanece no mesmo recorte, o contexto (cor do solo e iluminação) continua realista (Figura 3).

O desbalanceamento que resta (a pequena fração de pixels de formigueiro em cada imagem) é tratado pela função de custo combinada, apresentada a seguir.

3.2 Arquitetura U-Net

A escolha da U-Net deve-se ao fato de ser uma arquitetura consolidada e amplamente validada em segmentação semântica, eficaz para objetos pequenos e de contorno irregular e robusta em cenários com poucos exemplos rotulados, o que a torna uma linha de base apropriada e reprodutível para este primeiro estudo. A comparação com arquiteturas mais recentes e a aplicação de abordagens baseadas em detecção de objetos (em vez de segmentação semântica) estavam previstas, mas não puderam ser concluídas por restrições de tempo, permanecendo como trabalho futuro.

A arquitetura é uma U-Net totalmente convolucional, com um caminho de contração (codificador) e um de expansão (decodificador) ligados por conexões de salto (*skip connections*). A entrada é um recorte RGB de $3 \times 256 \times 256$. O encoder possui cinco estágios, com quatro reduções de resolução por *max-pooling* (agrupamento máximo) 2×2; a profundidade de canais evolui em $64 \rightarrow 128 \rightarrow 256 \rightarrow 256 \rightarrow 1024$, enquanto a resolução cai de 256×256 até 16×16 (gargalo). Cada estágio aplica um bloco convolucional duplo:

Figura 3: Aumento de dados por duplicação intra-recorte (quatro exemplos). Em cada par, à esquerda o recorte original e à direita o recorte aumentado, com as novas cópias do formigueiro destacadas em vermelho; cada cópia é inserida em uma nova posição, com rotação e espelhamento aleatórios.

duas convoluções 3×3 (padding 1), cada uma seguida de Normalização por Grupo (*Group Normalization*, 8 grupos) e ativação ReLU.

O decodificador possui quatro estágios simétricos. Em cada um, uma convolução transposta 2×2 (passo 2) dobra a resolução, o resultado é concatenado com o mapa de características do estágio correspondente do codificador (*skip connection*) e um bloco convolucional duplo refina a saída; a profundidade decresce $256 \to 256 \to 128 \to 64$. Uma convolução final 1×1 projeta para dois canais (fundo e formigueiro) na resolução original 256×256. A rede possui aproximadamente 19,7 milhões de parâmetros treináveis. A Tabela 2 detalha as camadas. As conexões de salto preservam detalhes finos de localização, importantes para segmentar regiões pequenas e de contorno irregular.

Tabela 2: Detalhamento das camadas da U-Net (entrada RGB). Bloco duplo $= 2\times$(Conv 3×3 pad 1 + GroupNorm(8) + ReLU); ConvT = convolução transposta 2×2 (passo 2); concat = concatenação com o *skip* do codificador.

Estágio	**Operação**	**Saída** ($C \times H \times W$)
Entrada	-	$3 \times 256 \times 256$
Enc1	bloco duplo	$64 \times 256 \times 256$
Enc2	MaxPool + bloco duplo	$128 \times 128 \times 128$
Enc3	MaxPool + bloco duplo	$256 \times 64 \times 64$
Enc4	MaxPool + bloco duplo	$256 \times 32 \times 32$
Gargalo	MaxPool + bloco duplo	$1024 \times 16 \times 16$
Dec1	ConvT + concat(Enc4) + bloco	$256 \times 32 \times 32$
Dec2	ConvT + concat(Enc3) + bloco	$256 \times 64 \times 64$
Dec3	ConvT + concat(Enc2) + bloco	$128 \times 128 \times 128$
Dec4	ConvT + concat(Enc1) + bloco	$64 \times 256 \times 256$
Saída	Conv 1×1	$2 \times 256 \times 256$

Além da normalização da entrada (média e desvio do ImageNet), descrita na preparação dos dados, a rede aplica normalização após cada convolução. Inicialmente utilizou-se Normalização por Lote (*Batch Normalization*) que, em conjunto com a superamostragem aprendida (*upsampling*) por convolução transposta (ConvTranspose2d) no decodificador, estabilizou o treinamento e foi decisiva para a evolução das métricas. Entretanto, devido à limitação de memória da GPU (6 GB), o treinamento emprega lotes pequenos (tamanho de lote igual a 2), regime no qual as estatísticas por lote da Normalização por Lote tornam-se instáveis. Por esse motivo, a Normalização por Lote foi substituída pela Normalização por Grupo (8 grupos), que normaliza grupos de canais de forma independente do tamanho do lote, preservando a estabilidade do treino. Como salvaguarda adicional contra explosão de gradientes, aplica-se recorte de gradiente (norma máxima 1,0).

Durante a inferência, a probabilidade da classe formigueiro é comparada a um limiar de confiança para gerar a máscara binária final.

3.3 Função de custo e otimização

Foram avaliadas diferentes estratégias de função de custo ao longo dos experimentos. A configuração de linha de base (*baseline*) utilizou exclusivamente Entropia Cruzada (*Cross-Entropy*), servindo como referência para avaliar o impacto incremental das demais estratégias. Em seguida, foram aplicadas combinações com a perda de Tversky [5], a perda Focal [3] e a perda de Lovász [6] para lidar melhor com o desbalanceamento de classes e aproximar o treinamento da métrica de IoU.

A perda de Tversky é definida por:

$$\mathcal{L}_T = 1 - \frac{TP}{TP + \alpha FP + \beta FN}, \tag{1}$$

em que TP, FP e FN representam verdadeiros positivos, falsos positivos e falsos negativos, respectivamente. Valores maiores de β penalizam mais os falsos negativos, favorecendo maior Revocação.

A perda Focal reduz a contribuição de exemplos fáceis e concentra o gradiente em pixels difíceis:

$$\mathcal{L}_F = -\sum_i (1 - p_{t,i})^\gamma \log(p_{t,i}). \tag{2}$$

A perda de Lovász [6] otimiza diretamente uma relaxação contínua do índice de Jaccard (IoU). No caso binário, define-se por pixel a margem $m_i = z_{i,1} - z_{i,0}$ (diferença entre os logits de formigueiro e de fundo) e o erro $e_i = 1 - m_i s_i$, com $s_i \in \{-1, +1\}$ indicando a classe verdadeira. Ordenando os erros de forma decrescente, a perda é

$$\mathcal{L}_{Lov} = \sum_i \max(0,\, e_{(i)})\, g_i, \tag{3}$$

em que $e_{(i)}$ são os erros ordenados e g é o gradiente da extensão de Lovász da perda de Jaccard, o que torna o IoU diferenciável.

A configuração final do modelo combina as três perdas:

$$\mathcal{L} = 0{,}5\,\mathcal{L}_T + 0{,}2\,\mathcal{L}_F + 0{,}3\,\mathcal{L}_{Lov}. \tag{4}$$

Adotaram-se $\alpha = 0{,}3$ e $\beta = 0{,}7$ na perda de Tversky (favorecendo a Revocação), $\gamma = 2{,}0$ na perda Focal e peso de classe $6{,}0$ para formigueiro (e $1{,}0$ para fundo) na componente Focal; os pixels não anotados (rótulo 255) são ignorados em todas as componentes.

O otimizador utilizado foi Adam [7], com taxa de aprendizado inicial de 1×10^{-3}. Também foi utilizado controle de gradiente para evitar instabilidades durante o treinamento.

3.4 Inferência e métricas de avaliação

Na etapa de inferência, a máscara final é obtida aplicando um limiar (*threshold*) sobre a probabilidade da classe formigueiro. Nos experimentos finais, o limiar de 0,40 foi utilizado para aumentar a Revocação. Esse limiar influencia diretamente o compromisso entre Revocação e Precisão. Limiares mais altos reduzem falsos positivos, porém aumentam falsos negativos. Limiares menores aumentam a capacidade de identificação, mas ampliam a quantidade de regiões incorretamente classificadas.

Para avaliação, foram usadas Precisão, Revocação e F1, que medem a capacidade do modelo de identificar corretamente a presença de formigueiros na imagem (Equação 5), e IoU e Dice, que avaliam a qualidade espacial da segmentação pixel a pixel (Equação 6). Nesse contexto, um modelo pode apresentar Revocação elevada, indicando boa capacidade de detecção, mas IoU moderado, refletindo limitações na precisão geométrica das máscaras preditas.

$$F1 = 2 \cdot \frac{Preciso \cdot Revocao}{Preciso + Revocao}. \tag{5}$$

$$IoU = \frac{TP}{TP + FP + FN}. \tag{6}$$

Foram conduzidas catorze rodadas de treinamento (Run 01 a Run 14), das quais sete são detalhadas na Tabela 7. Entre as rodadas, variaram-se quatro grupos de fatores:

- **Função de custo:** de Entropia Cruzada (linha de base) a combinações de Tversky, Focal e Lovász (Equação 4);
- **Aumento de dados:** espelhamento horizontal e vertical, rotação aleatória (± 15), variação fotométrica (brilho, contraste e saturação), deformação elástica ($\alpha = 25$, $\sigma = 4$) e técnicas específicas da classe positiva (colagem entre recortes, ou *copy-paste*, nas rodadas intermediárias e duplicação intra-recorte na configuração final);
- **Limiar de confiança:** avaliado em diferentes valores, adotando-se $0{,}40$ na configuração final (Tabela 3);
- **Filtragem de regiões conexas:** no pós-processamento, regiões muito pequenas ou muito grandes são descartadas; o tamanho mínimo passou de 100 px nas primeiras rodadas para 5 px na final, com máximo de 5.000 px.

As métricas foram calculadas sobre o conjunto de validação (2.466 imagens), considerando tanto a detecção por imagem (Precisão, Revocação e F1) quanto a segmentação pixel a pixel (IoU e Dice da classe formigueiro).

Os experimentos foram realizados em ambiente Linux Ubuntu, utilizando a biblioteca PyTorch para treinamento da rede neural. O treinamento foi executado em um computador com as seguintes especificações:

- Processador Intel Core i5-10400F;
- 16 GB de memória RAM;
- GPU NVIDIA GeForce RTX 2060 com 6 GB de VRAM.

4 Resultados e Discussão

Os experimentos foram organizados em rodadas (*runs*) numeradas, cada uma correspondendo a uma configuração de treinamento (arquitetura, função de custo, aumento de dados e pós-processamento), evoluídas de forma incremental a partir de uma linha de base com Entropia Cruzada. Esta seção destaca as rodadas que representam marcos dessa evolução, em especial a Run 05 (maior IoU) e a Run 14 (maior Revocação e menor número de falsos negativos), esta última adotada como configuração final por priorizar a detecção em cenários de monitoramento. As Tabelas 6 e 7 resumem, respectivamente, as configurações e os resultados das rodadas selecionadas.

4.1 Calibração do limiar e desempenho final

A Tabela 3 apresenta o impacto de diferentes valores de limiar.

Tabela 3: Impacto do limiar nas métricas de detecção.

Limiar	Precisão	Revocação	F1
0,50	84,2%	85,2%	84,7%
0,45	80,0%	87,7%	83,7%
0,40	75,9%	91,7%	83,1%
0,35	72,5%	94,1%	82,1%

O limiar é aplicado sobre a probabilidade da classe formigueiro: reduzi-lo classifica mais pixels como positivos, o que aumenta a Revocação (menos falsos negativos), mas também eleva os falsos positivos e reduz a Precisão. Esse efeito é monotônico na Tabela 3: ao baixar o limiar de 0,50 para 0,35, a Revocação sobe de 85,2% para 94,1% ($+8{,}9$ pp), enquanto a Precisão cai de 84,2% para 72,5% ($-11{,}7$ pp). O F1, por ser a média harmônica das duas, atinge o máximo em 0,50 (84,7%) e decresce à medida que o limiar diminui, pois o ganho de Revocação não compensa a perda de Precisão.

A escolha de 0,40 não corresponde, portanto, ao maior F1, mas a um compromisso deliberado em favor da detecção. No cenário de monitoramento, um falso negativo é operacionalmente mais custoso que um falso positivo: um formigueiro não detectado deixa de ser tratado, enquanto um falso alarme pode ser descartado em revisão manual. Com o limiar de 0,40, a Revocação alcança 91,7% (apenas 49 falsos negativos na Run 14), ao custo de cerca de 1,6 pp em F1 e 8 pp em Precisão frente a 0,50. Limiares ainda menores, como 0,35, elevam a Revocação a 94,1%, mas a Precisão de 72,5% gera um volume de falsos alarmes que torna 0,40 um equilíbrio mais adequado. A Figura 4 ilustra o efeito do limiar sobre uma mesma predição.

(a) $t = 0{,}50$ **(b)** $t = 0{,}40$ **(c)** $t = 0{,}35$

Figura 4: Efeito do limiar de confiança t sobre a mesma predição (região predita em vermelho). Limiares menores ampliam a área detectada (maior Revocação), ao custo de mais falsos positivos.

A Run 14 é a configuração final do trabalho (linha correspondente na Tabela 7): combina a perda Tversky + Focal + Lovász, a Normalização por Grupo (GroupNorm), a duplicação intra-recorte e o limiar $t = 0{,}40$, e foi escolhida por maximizar a detecção: maior Revocação e menor número de falsos negativos. A Tabela 4 apresenta seus resultados de detecção por imagem.

Tabela 4: Resultados de detecção por imagem na Run 14.

Métrica	Valor
TP	544
FP	173
FN	49
TN	1.700
Precisão	75,9%
Revocação	91,7%
F1	83,1%

A Tabela 5 apresenta os resultados pixel a pixel. Observa-se que a classe fundo obteve desempenho muito superior à classe formigueiro, reflexo direto do desbalanceamento dos dados.

Tabela 5: Resultados de segmentação pixel a pixel na Run 14.

Métrica	Valor
Acurácia por pixel	98,7%
IoU fundo	98,7%
IoU formigueiro	33,8%
Dice formigueiro	50,5%
mIoU global	66,3%
Dice médio global	74,9%

Embora a acurácia global seja elevada, ela não deve ser interpretada isoladamente, pois a classe fundo domina a maior parte dos pixels. Por isso, o IoU da classe formigueiro é uma métrica mais representativa da qualidade da segmentação.

4.2 Comparação entre rodadas

A Tabela 6 detalha a configuração de cada rodada selecionada (arquitetura/normalização, função de perda, estratégia de aumento de dados e limiar de confiança t), padronizando a nomenclatura empregada ao longo do texto; a Tabela 7 resume os resultados correspondentes. A seleção privilegia configurações representativas: a Run 03 (linha de base com Focal), a Run 05 (maior IoU), a Run 10 (melhor F1 equilibrado), as Runs 06, 08 e 11 (tentativas de melhoria que não superaram os marcos anteriores) e a Run 14 (maior Revocação, configuração final).

Distinguem-se duas estratégias de aumento da classe positiva: na **colagem entre recortes** (*copy-paste*), um formigueiro extraído de um recorte positivo é colado em outro recorte (em geral negativo); na **duplicação intra-recorte**, os formigueiros já presentes no recorte são replicados em novas posições do próprio recorte, preservando o contexto de solo e iluminação. A primeira foi empregada nas rodadas intermediárias (Runs 08, 10 e 11) e a segunda na configuração final (Run 14).

Tabela 7: Resultados (métricas) das rodadas selecionadas. As configurações correspondentes estão detalhadas na Tabela 6.

Run	FP	FN	Precisão	Revocação	IoU form.
03	433	133	51,5%	77,6%	23,9%
05	87	98	85,1%	83,5%	**35,2%**
06	79	112	**85,9%**	81,1%	34,3%
08	44	187	90,2%	68,5%	34,2%
10	105	88	82,8%	85,2%	30,3%
11	60	171	87,6%	71,2%	28,5%
14	173	**49**	75,9%	**91,7%**	33,8%

Os resultados mostram um compromisso entre segmentação precisa e maior capacidade de detecção. Configurações mais conservadoras reduzem falsos positivos e podem melhorar o IoU, mas deixam de detectar parte dos formigueiros. Por outro lado, reduzir o limiar aumenta a Revocação, mas também amplia falsos positivos em regiões visualmente semelhantes.

Para evidenciar a evolução incremental, a Tabela 8 reúne as rodadas que marcaram cada etapa. Como cada rodada alterou mais de um fator simultaneamente, os ganhos não podem ser atribuídos isoladamente a um único componente; ainda assim, a progressão evidencia o efeito agregado das melhorias. A linha de base com Entropia Cruzada não é incluída por ter colapsado para a classe de fundo, sem métricas úteis.

Tabela 6: Configurações das rodadas selecionadas, com nomenclatura padronizada. Aumento de dados geométrico = espelhamento horizontal/vertical + rotação aleatória ($\pm 15^{\circ}$); aumento de dados fotométrico = variação de brilho, contraste e saturação. CE = Entropia Cruzada (*Cross-Entropy*).

Run	Arquitetura / Normalização	Função de Perda	Estratégia de Aumento de Dados	Limiar t
03	U-Net base (bilinear, sem normalização)	Focal ($\gamma = 2{,}0$)	Geométrica + fotométrica	0,60
05	U-Net + BatchNorm + ConvTranspose2d	Tversky + CE (50/50)	Geométrica + fotométrica	0,60
06	U-Net + BatchNorm + ConvTranspose2d	Tversky + CE (70/30)	Geométrica + fotométrica + oversampling 3:1	0,50
08	U-Net + BatchNorm + ConvTranspose2d	Tversky + Focal (70/30)	Geométrica + fotométrica + colagem entre recortes (*copy-paste*, $p = 0{,}5$) + deformação elástica + rotação 90°	0,40
10	U-Net + BatchNorm + ConvTranspose2d	Tversky + Focal (85/15)	Geométrica + fotométrica + colagem entre recortes (*copy-paste*, $p = 0{,}1$) + deformação elástica	0,50
11	U-Net + BatchNorm + ConvTranspose2d	Tversky + Focal + Lovász (50/20/30)	Geométrica + fotométrica + colagem entre recortes (*copy-paste*, $p = 0{,}4$) + deformação elástica + filtro de recortes (>70% ignorados)	0,50
14	U-Net + GroupNorm + ConvTranspose2d	Tversky + Focal + Lovász (50/20/30)	Geométrica + fotométrica + duplicação intra-recorte ($p = 0{,}7$)	0,40

Tabela 8: Evolução incremental ao longo das rodadas que marcaram cada etapa.

Run	Principal mudança	Revocação	IoU form.
03	U-Net base; Focal	77,6%	23,9%
05	+ BatchNorm e ConvTranspose2d; Tversky + CE	83,5%	35,2%
10	+ Focal; colagem entre recortes (*copy-paste*); deformação elástica	85,2%	30,3%
14	+ Lovász; GroupNorm; duplicação intra-recorte; $t = 0{,}40$	**91,7%**	33,8%

Embora a Run 05 apresente o maior IoU (35,2%), a configuração final (Run 14) foi escolhida por priorizar a detecção: sua Revocação de 91,7% supera em 8,2 pp a da Run 05, com apenas 49 falsos negativos (ante 98). O IoU ligeiramente menor (33,8%) é um custo aceitável no monitoramento, em que deixar de detectar um formigueiro é mais grave que delimitá-lo com precisão imperfeita.

Vale destacar, contudo, que o maior ganho incremental observado (a passagem ao patamar de IoU em torno de 35%) coincidiu com a estabilização da arquitetura, quando foram introduzidas a normalização nas convoluções e a superamostragem aprendida (ConvTranspose2d). Como, nessa transição, a função de custo e a arquitetura foram alteradas de forma próxima, seus efeitos não podem ser plenamente isolados neste estudo, que utilizou uma única arquitetura. Mantida fixa a arquitetura nas etapas seguintes, as variações de desempenho passaram a ser governadas pelas estratégias de preparação de dados, escolha das funções de custo e calibração do limiar.

4.3 Análise Qualitativa

O modelo tem bom desempenho na maioria dos casos: formigueiros de tamanho médio a grande, isolados e com bom contraste em relação ao solo são detectados com alta confiança e máscaras bem delimitadas, o que se reflete na Revocação de 91,7%. O desempenho é mais consistente em solo escuro e uniforme, no qual a coloração do formigueiro se destaca do fundo. A Figura 5 apresenta exemplos de acerto, falso negativo e falso positivo.

Os falsos negativos ocorreram principalmente em três situações: formigueiros muito pequenos, regiões com textura semelhante ao solo e imagens com grande quantidade de área ignorada. Esses casos reduzem a confiança do modelo e dificultam a separação entre alvo e fundo.

Os falsos positivos foram associados principalmente a solo avermelhado, marcas de pneus, linhas de cultivo e sombras. Esses padrões apresentam textura e coloração semelhantes aos formigueiros, levando o modelo a classificar incorretamente algumas regiões como positivas.

Essa análise indica que novas anotações devem priorizar exatamente esses casos difíceis. A inclusão de mais exemplos de solo avermelhado sem formigueiro e de formigueiros em contextos variados tende a melhorar a capacidade de generalização do modelo.

5 Conclusão

Este trabalho apresentou um fluxo de processamento de segmentação semântica para detecção automática de formigueiros em ortofotos aéreas, baseado em U-Net e estruturado para lidar com forte desbalanceamento de classes. Os experimentos sustentam a hipótese de que melhorias sistemáticas na preparação de dados e no tratamento do desbalanceamento produzem ganhos substanciais na detecção e na segmentação da classe minoritária, partindo de uma linha de base com Entropia Cruzada que tendia a colapsar para a classe de fundo. Ressalta-se, contudo, que o maior salto de desempenho coincidiu com a estabilização da arquitetura (introdução de normalização nas convoluções e de superamostragem aprendida); como o estudo utilizou uma única arquitetura, os efeitos de dados e de arquitetura não puderam ser plenamente isolados, o que se registra como limitação e direção para trabalhos futuros.

Os resultados demonstraram que o modelo é capaz de detectar a maior parte dos formigueiros presentes no conjunto de validação, alcançando Revocação de 91,7% e F1 de 83,1%. Entretanto, a qualidade da segmentação pixel a pixel ainda é limitada pela baixa representatividade da classe formigueiro e pela semelhança visual com regiões de solo exposto.

A análise experimental mostrou que a acurácia global não é suficiente para avaliar o problema, pois o fundo domina a maior parte dos pixels. Métricas específicas da classe de interesse, como IoU e Dice do formigueiro, são mais

(a) Acerto

(b) Falso negativo

(c) Falso positivo

Figura 5: Casos qualitativos na validação. Em cada linha, da esquerda para a direita: imagem RGB, referência (*ground-truth*) e predição da Run 14 (região em vermelho). (a) acerto; (b) falso negativo (formigueiro não detectado); (c) falso positivo em solo avermelhado.

adequadas para medir a qualidade real da segmentação.

O trabalho também apresenta limitações. A baixa quantidade de exemplos positivos rotulados restringe a capacidade de generalização do modelo. Além disso, por utilizar apenas a arquitetura U-Net, sem comparação com modelos mais recentes (como DeepLabV3+ ou arquiteturas baseadas em Transformers) nem com abordagens baseadas em detecção de objetos, os efeitos da preparação de dados e os da arquitetura não puderam ser plenamente isolados; isolá-los exigiria reaplicar o mesmo fluxo de processamento de dados a arquiteturas distintas. Soma-se a isso o fato de os dados disponíveis se restringirem a imagens RGB de um único levantamento, já segmentadas em recortes de 256×256 pixels, sem bandas espectrais adicionais nem informação temporal. Por fim, não foram realizados testes em conjuntos externos ou em cenários operacionais reais de monitoramento agrícola.

Como trabalhos futuros, recomenda-se ampliar o conjunto de dados com novos exemplos positivos, incluir casos difíceis de falsos positivos e falsos negativos, testar redes-base (*backbones*) pré-treinadas, avaliar estratégias de anotação ativa, explorar métodos de segmentação de instâncias para estimar individualmente cada formigueiro detectado, avaliar imagens multiespectrais (por exemplo, com infravermelho próximo) para distinguir melhor vegetação e formigueiros, avaliar o efeito de recortes maiores (maior contexto espacial) sobre a detecção da classe minoritária e incorporar dados multitemporais que relacionem a detecção ao estágio de desenvolvimento dos formigueiros.

Declaração de uso de Inteligência Artificial

Os autores utilizaram o ChatGPT (OpenAI) exclusivamente como ferramenta de apoio à formatação, organização e revisão da redação deste manuscrito. O desenvolvimento científico do trabalho, incluindo concepção, metodologia, implementação, experimentos, análise dos resultados e conclusões, é de responsabilidade exclusiva dos autores.

Referências

[1] Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[2] Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In: *Medical Image Computing and Computer-Assisted Intervention*, 2015.

[3] Lin, T.-Y. et al. Focal loss for dense object detection. In: *IEEE International Conference on Computer Vision*, 2017.

[4] Milletari, F.; Navab, N.; Ahmadi, S.-A. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In: *International Conference on 3D Vision*, 2016.

[5] Salehi, S. S. M.; Erdogmus, D.; Gholipour, A. Tversky loss function for image segmentation using 3D fully convolutional deep networks. In: *Machine Learning in Medical Imaging*, 2017.

[6] Berman, M.; Triki, A. R.; Blaschko, M. B. The Lovász-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[7] Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. In: *International Conference on Learning Representations*, 2015.

[8] Buslaev, A. et al. Albumentations: Fast and flexible image augmentations. *Information*, v. 11, n. 2, 2020.

[9] Kamilaris, A.; Prenafeta-Boldú, F. X. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, v. 147, p. 7090, 2018.

[10] Mohanty, S. P.; Hughes, D. P.; Salathé, M. Using deep learning for image-based plant disease detection. *Frontiers in Plant Science*, v. 7, art. 1419, 2016.

[11] Milioto, A.; Lottes, P.; Stachniss, C. Real-time semantic segmentation of crop and weed for precision agriculture robots leveraging background knowledge in CNNs. In: *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[12] Sa, I. et al. WeedMap: A large-scale semantic weed mapping framework using aerial multispectral imaging and deep neural network for precision farming. *Remote Sensing*, v. 10, n. 9, art. 1423, 2018.

[13] Della Lucia, T. M. C.; Gandra, L. C.; Guedes, R. N. C. Managing leaf-cutting ants: peculiarities, trends and challenges. *Pest Management Science*, v. 70, n. 1, p. 1423, 2014.

[14] Tsouros, D. C.; Bibi, S.; Sarigiannidis, P. G. A review on UAV-based applications for precision agriculture. *Information*, v. 10, n. 11, art. 349, 2019.

[15] dos Santos, A. et al. Remote detection and measurement of leaf-cutting ant nests using deep learning and an unmanned aerial vehicle. *Computers and Electronics in Agriculture*, v. 198, art. 107071, 2022.

[16] Santos, I. C. L. et al. Remote sensing to detect nests of the leaf-cutting ant Atta sexdens (Hymenoptera: Formicidae) in teak plantations. *Remote Sensing*, v. 11, n. 14, art. 1641, 2019.