UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Classificação automática de imagens de satélite para acompanhamento e controle do desmatamento na Amazônia

Matheus de Castro Quirino

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP SÃO CARLOS

Matheus de Castro Quirino

Classificação automática de imagens de satélite para acompanhamento e controle do desmatamento na Amazônia

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Fernando Pereira dos Santos

Versão original

São Carlos

2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi, ICMC/USP, com os dados fornecidos pelo(a) autor(a)

Quirino, Matheus de Castro
S856m Classificação automática de imagens de satélite para acompanhamento e controle do desmatamento na Amazônia / Matheus de Castro Quirino ; orientador Fernando Pereira dos Santos. – São Carlos, 2023.
59 p.

Monografia (MBA em Inteligência Artificial e Big Data) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2023.

1. Classificação multirrótulo. 2. Classificação multi-classe. 3. Aprendizado profundo. 4. Visão computacional. I. Dos santos, Fernando P., orient. II. Título.

Matheus de Castro Quirino

Automatic classification of satellite images for monitoring and controlling deforestation in the Amazon rainforest

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Prof. Dr. Fernando Pereira dos Santos

Original version

São Carlos

2023

RESUMO

Quirino, M. C. **Classificação automática de imagens de satélite para acompanhamento e controle do desmatamento na Amazônia**. 2023. 59p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2023.

Buscando contribuir para atividades de acompanhamento e controle do desmatamento da Amazônia, o trabalho construiu, aplicando distintas técnicas de *deep learning*, modelos de visão computacional voltados à classificação de rótulos de condições atmosféricas e de elementos de superfície presentes em imagens de satélite. Ao automatizar a classificação, esse modelo pode tanto ser utilizado para um acompanhamento em tempo real das áreas com ações antropogênicas, quanto como uma ferramenta de suporte para a atuação do fotointerprete. Para desenvolver os modelos, o trabalho utilizou um conjunto com cerca de 40 mil imagens disponibilizadas no *Kaggle* em 2017. Consultadas essas imagens, o estudo buscou entender as características dos rótulos a serem preditos, avaliando a distribuição das ocorrências dessas classes e analisando as correlações entre elas. Nessa etapa, verificou-se que as imagens abrangiam 17 rótulos, divididos em condições atmosféricas e elementos comuns e raros de superfície. Na modelagem, o estudo evoluiu gradualmente a complexidade dos desenvolvimentos, explorando técnicas de *transfer learning*, aumentação de dados e variações nos parâmetros do modelo. Finalizados os desenvolvimentos, a melhor abordagem foi composta combinando 2 modelos, sendo um associado à predição multi-classe das condições atmosféricas e o outro à predição multirrótulo dos elementos de superfície. Considerando o cenário combinado, a performance alcançada foi de 0,918 na métrica F_β, selecionada para lidar com o desbalanceamento dos rótulos. Avaliando as classes individualmente, para as condições atmosféricas, o modelo conseguiu classificar imagens limpas ou parcialmente nubladas com performances de 0,970 e 0,925, mas apresentou dificuldade para classificar imagens nubladas (0,645). Para elementos de superfície, influenciado pelo desbalanceamento das classes no conjunto de treino, o modelo apresentou bons resultados para rótulos como vegetação primária (0,99), agricultura (0,894) e estrada (0,870), mas teve dificuldade ao classificar elementos pouco frequentes, como regiões de mineração (0,485), extração seletiva (0,396) e solo exposto (0,311). Por esses resultados, o trabalho concluiu que o objetivo proposto de construir uma modelagem capaz de classificar automaticamente imagens de satélite da Amazônia foi atendido, no entanto avaliou que existem margens de melhoria, sobretudo nas classes menos presentes nos dados selecionados.

Palavras-chave: Classificação multirrótulo. Classificação multi-classe. Aprendizado profundo. Visão computacional.

ABSTRACT

Quirino, M. C. **Automatic classification of satellite images for monitoring and controlling deforestation in the Amazon rainforest**. 2023. 59p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2023.

Seeking to contribute to activities related to monitoring and controlling deforestation in the Amazon rainforest, this monograph constructed computer vision models using various deep learning techniques, with a focus on classifying multiple labels of atmospheric conditions and surface elements present in satellite images. By automating the classification, this work can be applied for real-time monitoring of areas with anthropogenic activities and as a support tool for photointerpreters' tasks. The study used a dataset with approximately 40,000 images available on Kaggle in 2017 to develop the models. Upon examining these images, the work aimed to comprehend the characteristics of their labels by evaluating the distribution of these classes and analyzing the correlations among them. At this stage, it was observed that the images encompassed 17 classes, categorized into atmospheric conditions, common surface elements, and rare surface elements. During the modeling phase, the study gradually increased the complexity of the developments, exploring techniques such as transfer learning, data augmentation, and variations in model parameters. After completing the development tasks, the best approach was composed by combining two models, with one dedicated to multi-class prediction of atmospheric conditions and the other to multi-label prediction of surface elements. Considering the combined scenario, the achieved performance was 0.918 on the F-beta metric, selected to address label imbalance. When evaluating the classes individually for atmospheric conditions, the model was able to classify clear or partly cloudy images with performances of 0.970 and 0.925, but it had difficulty classifying cloudy images (0.645). Regarding surface elements, influenced by the imbalance of labels in the training set, the model achieved good results for classes such as primary vegetation (0.99), agriculture (0.894), and road (0.870), but encountered difficulties when classifying less frequent elements such as conventional mining (0.485), selective logging (0.396), and bare ground (0.311). Based on these results, the study concluded that the proposed objective of constructing a model capable of automatically classifying satellite images of the Amazon rainforest was achieved. However, it also assessed that there are opportunities for improvement, especially within the less-represented classes in the selected data.

Keywords: Multi-label classification. Multi-class classification. Deep Learning. Computer vision.

LISTA DE FIGURAS

Figura 1 – Exemplo do processo de convolução 29
Figura 2 – Arquitetura - VGG16 30
Figura 3 – Sequência de atividades 33
Figura 4 – Ocorrências dos rótulos de condições atmosféricas 37
Figura 5 – Exemplos de imagens para rótulos de condições atmosféricas 38
Figura 6 – Distribuição dos rótulos para os elementos de superfície 38
Figura 7 – Histogramas da quantidade de rótulos por imagem 39
Figura 8 – Matriz de correlação entre os elementos de superfície 40
Figura 9 – Exemplos de imagens para rótulos de superfície 41
Figura 10 – Resultados dos modelos para o cenário 1 das modelagens iniciais 43
Figura 11 – Curvas de aprendizado dos modelos 43
Figura 12 – Resultados dos modelos iniciais para as configurações avaliadas 44
Figura 13 – Exemplos de imagens *artisinal mine* com classificação invertida 46
Figura 14 – Exemplos de imagens *bare ground* e *cultivation* com classificação invertida 47
Figura 15 – Exemplos de imagens *haze* classificadas como *clear* 48
Figura 16 – Curvas de aprendizado contendo, ou não, variação no *learning rate* . . 49
Figura 17 – Resultados dos modelos iniciais para as configurações avaliadas 50
Figura 18 – Curva de aprendizado para a ResNet50 com aumentação dos dados . . 50
Figura 19 – Resultados da ResNet50 com a variação nas dimensões das imagens . . 51
Figura 20 – Tempos de treinamento em função das dimensões dos *inputs* 51
Figura 21 – Resultados da abordagem por modelos segregados 52
Figura 22 – Resultados variando o *threshold* de classificação dos elementos de superfície 53
Figura 23 – Exemplos de imagens com classificações corrigidas no modelo final . . . 55

LISTA DE TABELAS

Tabela 1 – Arquitetura da CNN definida por experimentação 42
Tabela 2 – Detalhamento do resultado por rótulo 45
Tabela 3 – Classes reais e preditas para imagens com rótulo *artisinal mine* 46
Tabela 4 – Classes reais e preditas para imagens com rótulo *bare ground* 46
Tabela 5 – Classes reais e preditas para imagens com rótulo *cultivation* 47
Tabela 6 – Classes reais e preditas para imagens com rótulo *haze* 48
Tabela 7 – Detalhamento do resultado por rótulo para o modelo final 54
Tabela 8 – Comparativo das classificações de condições atmosféricas 55

SUMÁRIO

1 INTRODUÇÃO 17
1.1 Motivação 17
1.2 Objetivo 19
1.3 Estrutura do trabalho 19

2 FUNDAMENTAÇÃO TEÓRICA 21
2.1 Contextualização da área de Visão Computacional 21
2.2 *Deep Learning* 22
2.2.1 *Feedforward neural networks* 23
2.2.2 Algoritmos de otimização 24
2.2.3 *Backpropagation* 25
2.2.4 Funções de ativação 25
2.2.5 Funções de custo 27
2.3 *Convolutional Neural Networks* 28

3 METODOLOGIA 33
3.1 Caracterização do problema 33
3.2 Modelagens iniciais 34
3.3 Modelagens finais 35

4 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS 37
4.1 Análise exploratória 37
4.2 Resultados dos modelos iniciais 42
4.3 Resultados dos modelos finais 48

5 CONCLUSÕES 57

Referências 59

1 INTRODUÇÃO

Nesse primeiro capítulo, apresentam-se a motivação e os objetivos do estudo e descreve-se a estrutura do trabalho a ser explorada nas seções seguintes.

1.1 Motivação

O desmatamento da Amazônia, entendido como a supressão de áreas de vegetação primária por ações antropogênicas, que já se encontrava em patamares críticos, está avançando de forma acelerada nos últimos anos, o que intensifica a relevância de soluções que contribuam para um controle eficiente e estratégico da região. Segundo dados do Programa de Monitoramento da Floresta Amazônica Brasileira por Satélite (PRODES), desenvolvido pelo Instituto de Pesquisas Espaciais (INPE), as taxas anuais de desmatamento da floresta, avaliadas em torno de 7 mil km² até 2018, estão superiores a 10 mil km² desde 2019 e atingiram um pico de 13 mil km² no ano de 2021, o que seria equivalente a 8,5 vezes o município de São Paulo. No total, a região desmatada já ultrapassou 830 mil km², valor superior a 3,4 vezes a área do estado de São Paulo e que corresponde a 17% da Amazônia Legal, que, por sua vez, abrange 59% do território brasileiro (INPE, 2023).

As causas desse avanço do desmatamento são diversas e complexas, podendo-se destacar a expansão agropecuária, a realização de obras de mineração, estradas e barragens, e a impunidade a crimes ambientais. Além desses aspectos, que contribuem para uma modificação intensa da cobertura vegetal em um curto intervalo de tempo, naquele que é classificado como desmatamento por corte raso, a Amazônia também é ameaçada pelo processo de degradação progressiva, em que ocorre uma gradativa perda da qualidade original da floresta, causada por fatores como exploração madeireira, caças, queimadas e eventos climáticos. Como consequência, a combinação desses cenários de desmatamento provoca efeitos como perda de biodiversidade em larga escala, tanto da fauna quanto da flora local, desequilíbrio ambiental, danos à segurança alimentar da população e, também, intensificação de mudanças climáticas e de eventos climáticos extremos (INPE, 2022).

Diante desse contexto de expansão nos níveis de desmatamento e dada a criticidade dos impactos atrelados, nota-se a relevância de soluções, como o PRODES e o Sistema de Detecção de Desmatamento em Tempo Real (DETER), também desenvolvido pelo INPE, que contribuam para o monitoramento eficiente dos locais ameaçados. No caso do PRODES, programa iniciado em 1988, o principal objetivo é estimar a taxa anual de desmatamento da floresta primária na Amazônia Legal Brasileira. Para isso, por meio da fotointerpretação por especialistas de imagens de satélite com resolução espacial entre 20 e 30 metros, o projeto realiza o mapeamento anual dos incrementos do desmatamento, classificando-os em grupos de desmatamento por corte raso ou por degradação progressiva

da vegetação. Já o DETER, por meio de alertas gerados diariamente, possui como principal objetivo informar rapidamente os órgãos de fiscalização sobre novas alterações na cobertura florestal. Com esses alertas, que também são disponibilizados para consulta pública no portal TerraBrasilis, os órgãos de fiscalização conseguem, de forma ágil, planejar suas ações e atuar no controle da região (INPE, 2022).

Até 2015, o DETER utilizava imagens geradas por sensores com resolução espacial de 250 metros, que permitia gerar avisos para alterações da cobertura vegetal com área mínima de 25 hectares, mas não possibilitava a classificação das imagens em rótulos mais detalhados. A partir de 2015, a metodologia do DETER foi aprimorada e passou a utilizar imagens com resolução espacial entre 56 e 64 metros, o que viabilizou reduzir a área mínima para 3 hectares e permitou um maior detalhamento dos alertas, que passaram a ser classificados nos casos de desmatamento com solo exposto ou com vegetação, mineração, degradação, cicatriz de incêndio florestal e cortes seletivos de exploração madeireira (desordenado ou geométrico). Para executar essas classificações, a metodologia utilizada pelo DETER baseia-se na fotointerpretação das imagens a partir dos elementos de tonalidade, cor, forma, textura e contexto. Os fotointérpretes avaliam, também, os resultados do Modelo Linear de Mistura Espectral (MLME), que é responsável por estimar as frações de solo, vegetação e sombra presentes em cada imagem (INPE, 2022).

Além dos métodos adotado pelo DETER e pelo PRODES, robustos pelo conhecimento especialista do fotointérprete, outra abordagem aplicável para a classificação das imagens da Amazônia consiste no uso de técnicas de *Deep Learning*, por meio do treinamento de *Convolutional Neural Networks* (CNNs). Nesse caso, os modelos treinados em imagens de satélite previamente rotuladas podem ser utilizados para classificar novas imagens coletadas. Essa alternativa, por automatizar a classificação, pode ser utilizada tanto para um acompanhamento em tempo real das áreas com ações antropogênicas, quanto como uma informação adicional para auxiliar e aprimorar a atuação do fotointerprete. No presente trabalho, essa foi a abordagem implementada. Em particular, o estudo procurou desenvolver e comparar modelos de CNNs que, dadas imagens de satélite da Amazônia, fossem capazes de avaliá-las automaticamente com um elevado nível de assertividade, classificando tanto as condições atmosféricas quanto os diferentes elementos da superfície terrestre presentes nelas, sejam naturais ou resultantes de ações antropogênicas.

1.2 Objetivo

Dado o contexto apresentado no tópico anterior, o trabalho possui como objetivo geral desenvolver um modelo que, fornecida uma imagem de satélite da região amazônica, realize classificações multirrótulos dessa imagem.

Em termos específicos, esse objetivo pode ser detalhado nas seguintes questões:

- Qual é a condição atmosférica existente em uma imagem selecionada (totalmente nublado, parcialmente nublado, céu limpo, neblina)?
- Quais são os elementos naturais (florestas primárias, rios, lagos) e resultantes de ações antropogênicas (estradas, regiões de cultivo, habitação, mineração, garimpo, queimadas) existentes em uma imagem selecionada?
- Como a aplicação de distintas técnicas de *Deep Learning*, como o uso de redes pré-treinadas e a realização de aumentação nos dados, podem contribuir para a melhoria nos resultados das classificações?

1.3 Estrutura do trabalho

Em relação à estrutura do trabalho, após o atual capítulo de introdução, apresenta-se a fundamentação teórica, na qual são explorados os aspectos conceituais dos métodos de Visão Computacional utilizados no estudo. Na sequência, o terceiro capítulo delimita o escopo do problema avaliado e descreve a metodologia a ser implementada para a abordagem desse problema. O quarto capítulo, por sua vez, detalha os principais resultados obtidos. Por fim, o quinto capítulo apresenta as conclusões do trabalho, assim como discute oportunidades de melhorias e de novos desenvolvimentos que podem ser explorados em projetos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Para a seção de fundamentos teóricos, o primeiro tópico contempla uma breve contextualização da área de Visão Computacional, utilizando como referência, sobretudo, os estudos de Szeliski (2022) e Davies (2017). Após isso, utilizando como referência os livros de Goodfellow, Bengio e Courville (2016), Chollet (2021) e Géron (2022), os próximos tópicos detalham conceitos gerais das abordagens de *deep learning* e, em seguida, aprofundam nas explicações de elementos das *convolutional neural networks* (CNNs) e das arquiteturas de CNNs utilizadas durante as etapas de modelagem do estudo.

2.1 Contextualização da área de Visão Computacional

Como descrito por Szeliski (2022) e por Davies (2017), a Visão Computacional é um campo da Ciência da Computação direcionado ao desenvolvimento de ferramentas computacionais que sejam capazes de analisar, interpretar e extrair automaticamente informações do mundo a partir de imagens ou sequências de imagens. Contemplando aplicações diversas, as soluções de Visão Computacional estão sendo aplicadas, dentre outros problemas, para tarefas de classificação e segmentação de imagem, detecção de objetos, análises e predições de movimentos e reconstrução tridimensional de objetos a partir de múltiplas imagens (FORSYTH; PONCE, 2011; DAVIES, 2017).

Os primeiros estudos em visão computacional ocorreram no início da década de 70, estimulados pelo interesse do período nas áreas inteligência artificial e robótica. Nessa época, o que diferenciou a visão computacional do campo já existente de processamento digital de imagens foi o desejo de recuperar estruturas 3D de objetos a partir de feições 2D analisadas. Na década de 80, a atenção passou a ser direcionada, sobretudo, para o avanço em técnicas matemáticas mais sofisticadas e focadas na análise quantitativa das imagens. Já nos anos 90, alguns temas que começaram a adquirir relevância foram análise de movimentação de objetos, segmentação de imagens, reconhecimento de faces e computação gráfica. Nos anos 2000, dentre os temas que ganharam destaque, mencionam-se os algoritmos de reconhecimento de imagens baseados em *features*, os modelos de combinação de imagens para criação de texturas e os modelos de geração de superfícies tridimensionais mais realista. Nesse período, como consequência da maior disponibilidade de bases rotuladas na internet, houve, ainda, o início do crescimento na aplicação de técnicas de *machine learning* em problemas de visão computacional (SZELISKI, 2022).

Já a partir de 2010, houve uma intensificação profunda dos avanços na área de visão computacional, com os algoritmos sendo aprimorados tanto em termos de performance quando de robustez e confiabilidade, o que permitiu que eles passassem a ser utilizados de forma ampla em soluções comerciais. Durante esse período, os elementos que contribuíram

para o avanço foram diversos e potencializaram-se de maneira conjunta. Um primeiro fator consistiu na disponibilização de bases de dados anotados de larga escala, como a *ImageNet*. Conjuntamente a isso, houve um intenso aumento no poder computacional, viabilizado pela aplicação paralelizada dos algoritmos de aprendizado nas unidades de processamento gráfico (GPUs). Combinados, o acesso a dados em larga escala e a disponibilidade de recursos computacionais para processar esses dados permitiram que modelos robustos de *deep learning* fossem construídos e continuamente aprimorados (CHOLLET, 2021).

Como consequência, esses métodos de modelagem, que serão detalhados no tópico seguinte, potencializaram a transformação das soluções tanto nas tarefas de reconhecimento de imagens, que é o escopo do presente trabalho, quanto nas outras aplicações de visão computacional, e passaram a ser utilizados de forma geral nos desenvolvimentos da área. Além disso, viabilizaram que plataformas completas e complexas, envolvendo, por exemplo, veículos autônomos, realidades aumentadas e mapeamento e localização de objetos tridimensionais em tempo real pudessem ser implementadas (SZELISKI, 2022).

2.2 *Deep Learning*

Conceitualmente, *deep learning* consiste em uma subárea de *machine learning*, com abas contemplando o processo de aprendizado, no qual o modelo é treinado para, a partir dos *inputs* fornecidos, determinar a melhor representação dos dados capaz de gerar os *outputs* desejados (CHOLLET, 2021). Por outro lado, o que particulariza as abordagens de *deep learning* é que, para elas, esse aprendizado é determinado pela sobreposição de múltiplas camadas de representação. Com essas sobreposições, cada nova camada torna-se capaz de computar conceitos progressivamente mais complexos, combinados das representações mais simples geradas nas camadas anteriores (GOODFELLOW; BENGIO; COURVILLE, 2016). Exemplificando esse conceito para tarefas de reconhecimento de imagens, a partir dos *pixels* fornecidos como inputs iniciais, os modelos de *deep learning* são capazes, por exemplo, de gerar representações de linhas e bordas, que, em seguida, podem ser combinadas em contornos, formas e assim sucessivamente, até que o entendimento final da imagem seja alcançado.

Em *deep learning*, o aprendizado dessa estrutura hierárquica de representação de dados é viabilizado pelas *neural networks*. Como descrito por Chollet (2021), elas consistem em modelos construídos pela sobreposição de camadas de funções matemáticas, com cada uma dessas camadas sendo responsável por aplicar a transformação de dados que resultará na representação de dados associada. Nesse processo, os pesos de cada camada parametrizam a transformação a ser aplicada, e o aprendizado do modelo consiste na busca da melhor configuração desses pesos, de tal forma que os *outputs* finais sejam, o mínimo possível, divergentes dos *targets* desejados. Para mensurar essa divergência entre predições e valores reais, os modelos de *deep learning* se baseiam no cálculo de uma função

de custo, selecionada conforme a natureza do problema modelado. E, por meio do processo de *backpropagation*, essa função de custo é utilizada como *feedback* para o ajuste dos pesos das camadas, que, progressivamente, são corrigidos para minimizar o erro associado.

Partindo do contexto anterior, os próximos tópicos aprofundam em conceitos de *deep learning* relevantes para o desenvolvimento do trabalho. Inicialmente, apresentam-se as *feedforward neural networks*, abordagem base para os modelos de *deep learning*, com fundamentos válidos para diversas implementações, dentre as quais as CNNs. Em seguida, aprofunda-se na explicação do processo de treinamento dos modelos, introduzindo alguns dos algoritmos de otimização aplicados, detalhando o método de *backpropagation* e descrevendo algumas das funções de ativação e de custo usualmente utilizadas.

2.2.1 *Feedforward neural networks*

Assim como descrito de uma forma mais ampla para os modelos de *deep learning*, as *feedforward neural networks* são geradas pela sobreposição de camadas de funções matemáticas, compondo uma função final f que aprende o melhor conjunto de pesos $\boldsymbol{W}$ para se aproximar da função f^*, que associa um determinado *input* $\mathbf{x}$ a um *output* $\mathbf{y}$ (GOODFELLOW; BENGIO; COURVILLE, 2016). Para um cenário com três camadas, por exemplo, esse comportamento pode ser representado pela expressão $f(\mathbf{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x})))$, em que $f^{(1)}$ corresponde à primeira camada, $f^{(2)}$ à segunda e assim sucessivamente.

Cada camada, além disso, pode ser composta por múltiplas unidades, que atuam de forma paralela e determinam os *outputs* dessa camada. Para isso, em cada unidade, os *inputs* recebidos são somados linearmente, sendo ponderados pelos seus respectivos pesos, o que pode ser representada por $z = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b = \mathbf{w}^T\mathbf{x} + b$, sendo $\mathbf{w}$ o vetor de pesos, $\mathbf{x}$ o vetor de *inputs* e b o termo *bias* da soma. Ao resultado dessa combinação, aplica-se uma função de ativação não linear, que é responsável por gerar o *output* da unidade, como representado por $h_{\mathbf{w}}(\mathbf{x}) = \phi(\mathbf{w}^T\mathbf{x} + b)$, em que ϕ corresponde à função de ativação aplicada (GÉRON, 2022). É possível, ainda, expandir essa expressão para abranger todas as unidades da camada, como representado por $\mathbf{h}_{\mathbf{W},\mathbf{b}}(\mathbf{X}) = \phi(\mathbf{XW} + \mathbf{b})$, em que $\mathbf{W}$, $\mathbf{X}$, $\mathbf{b}$ e $\mathbf{h}_{\mathbf{W},\mathbf{b}}$ correspondem, respectivamente, às matrizes de pesos e *inputs*, e aos vetores de *bias* e *outputs* da camada. Definido esse vetor $\mathbf{h}_{\mathbf{W},\mathbf{b}}(\mathbf{X})$ para a camada, ele pode, em seguida, ser enviado como *input* para a próxima, repetindo, em cadeia, o processo descrito até que os *outputs* finais sejam obtidos.

Para as *feedforward neural networks*, esse fluxo é sempre executado, até a predição, com as camadas sendo percorridas de forma acíclica e direcionada. Justamente por conta disso, receberam a nomenclatura *feedforward*. Esse comportamento é diferente, por exemplo, das *recurrent neural networks*, que passam a contemplar conexões de *feedback*, nas quais os *outputs* de uma camada podem ser retroalimentados como *inputs* dessa própria camada para as predições seguintes (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.2.2 Algoritmos de otimização

Para que o processo de aprendizado seja alcançado, um dos componentes chave em *deep learning* consiste no algoritmo de otimização implementado, que torna possível atualizar os pesos do modelo de forma iterativa, minimizando a função de custo associada. De uma forma geral, os principais algoritmos utilizados possuem o método *gradient descent* como base, tendo sido aprimorados a partir dele para tornar a otimização mais rápida e eficiente (GÉRON, 2022).

Para o entendimento desse método, pode-se considerar, inicialmente, uma função $y = f(x)$. Dada essa função, sua derivada $f'(x)$ permite determinar como pequenas alterações de tamanho ϵ no *input* geram a correspondente variação no *output*, o que pode ser representado por $f(x+\epsilon) \approx f(x)+\epsilon f'(x)$. Partindo desse cenário, como apresentado por Goodfellow, Bengio e Courville (2016), o método *gradient descent* consiste na minimização de $f(x)$ pelo deslocamento de x, em pequenos passos, na direção oposta à derivada, sendo que, no cenário com múltiplos atributos, esse método é aplicado pela movimentação na direção oposta ao gradiente associado. Nesse caso, representado por $\boldsymbol{\nabla}_{\mathbf{x}} f(\mathbf{x})$, o gradiente é o vetor composto pelas derivadas parciais $\frac{\partial}{\partial x_i} f(\mathbf{x})$, em que cada uma delas mensura o quanto f é alterada quando cada um dos atributos x_i são variados em $\mathbf{x}$. Considerando esses conceitos, o método *gradient descent* pode ser representado por $\mathbf{x}' = \mathbf{x} - \epsilon \boldsymbol{\nabla}_{\mathbf{x}} f(\mathbf{x})$, sendo $\mathbf{x}'$ os novos valores dos *inputs* após a execução do passo de tamanho ϵ, usualmente denominado *learning rate*.

Aplicado aos modelos de *deep learning*, o método anterior é utilizada para otimizar os pesos $\mathbf{W}$ minimizando a função de custo $L(\mathbf{W})$, avaliada para cada um dos m registros do conjunto de treinamento. Nesse cenário, a atualização dos pesos é representada por:

$$\mathbf{W}' = \mathbf{W} - \epsilon \frac{1}{m} \sum_{i=1}^{m} \boldsymbol{\nabla}_{\mathbf{W}} L(\mathbf{W}; \mathbf{x}^{(i)}) \tag{2.1}$$

A necessidade de computar a função de custo para todos os registros de treinamento torna a abordagem anterior lenta, principalmente quando volume de dados é grande. Aprimorando esse processo, a variação *stochastic gradient descent* (SGD) seleciona, em cada atualização dos pesos, uma amostra aleatória $\mathbb{B}$, de tamanho m', do conjunto de treinamento, sendo a amostra $\mathbb{B}$ e o hiperparâmetro m' denominados, respectivamente, *minibatch* e *batch size*. Para o SGD, a expressão (2.1) passa a ser representada por (GOODFELLOW; BENGIO; COURVILLE, 2016):

$$\mathbf{W}' = \mathbf{W} - \epsilon \frac{1}{m'} \sum_{i=1}^{m'} \boldsymbol{\nabla}_{\mathbf{W}} L(\mathbf{W}; \mathbf{x}^{(i)}) \tag{2.2}$$

Além do SGD, o método *Adam*, nome derivado de *adaptive moment estimation*, é aplicado, também, de forma recorrente para a otimização dos pesos dos modelos. Além

de contemplar os benefícios do SGD, o método *Adam* é capaz de proporcionar uma adaptação individualizada do *learning rate* para cada um dos pesos, o que contribui para uma convergência mais rápida durante o treinamento e evita oscilações indesejadas em regiões próximas ao mínimo. Para isso, calcula iterativamente dois atributos, denominados momentos de primeira e de segunda ordem, que determinam a média e a variância dos gradientes e são avaliados por meio de médias móveis com taxas de decaimento exponencial (GÉRON, 2022).

2.2.3 *Backpropagation*

Aplicado de forma integrada ao algoritmo de otimização, o método *backpropagation* permite que, em cada passo durante o treinamento, sejam calculados os gradientes da função de custo em relação a cada peso do modelo. A partir disso, esses gradientes podem ser utilizados pelo algoritmo de otimização para realizar a atualização dos pesos, seguindo a abordagem descrita no tópico anterior (GOODFELLOW; BENGIO; COURVILLE, 2016).

A primeira etapa no método *backpropagation*, denominada *forward pass*, consiste em seguir o fluxo de informação descrito para as *feedforward neural networks* em 2.2.1, partindo dos *inputs* e percorrendo as camadas intermediárias até a obtenção dos *outputs* da camada de predição. Ao final dessa etapa, calcula-se o erro geral da rede pela função de custo definida para o modelo. Em seguida, inicia-se o próximo estágio, denominado *backward pass*. Nele, aplicando o conceito de regra da cadeia definido em Cálculo, determina-se, inicialmente, o quanto cada conexão da camada final contribuiu para o erro de cada *output* gerado. Finalizado esse estágio, o processo apresentado é repetido retroativamente para as conexões das camadas anteriores, até as conexões com os *inputs* iniciais (GÉRON, 2022).

Com essa abordagem, consegue-se propagar o gradiente de erro para cada peso e termo *bias* do modelo, permitindo que, com esses gradientes conhecidos, seja aplicado o passo de otimização descrito em 2.2.2. Durante o treinamento, a execução desses passos, contemplando as etapas de *backpropagation*, aplicação do algoritmo de otimização e atualização dos pesos do modelo é realizada para cada um dos *minibatchs*. Após todos os *minibatchs* terem sido avaliados, diz-se que uma época foi concluída. Pode-se, a partir disso, repetir o processo por tantas épocas quanto forem necessárias, até que a métrica de avaliação do modelo alcance seu estágio ideal.

2.2.4 Funções de ativação

Como descrito para as *feedforward neural networks*, as funções de ativação são aplicadas em cada unidade das camadas de transformação dos dados, gerando o *output* da unidade, representado por $h_{\mathbf{w}}(\mathbf{x}) = \phi(\mathbf{w}^T\mathbf{x} + b)$. Além disso, sobretudo para as camadas intermediárias, são não lineares, o que viabiliza que o modelo consiga se aproximar de uma função f^*, mesmo que ela apresente comportamento complexo e não linear. Caso contrário,

esse modelo estaria restrito à representação de funções lineares, ainda que contemplasse múltiplas camadas (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para as unidades das camadas intermediárias, a *rectified linear unit* (ReLU), expressa por $ReLU(z) = max\{0,z\}$, consiste em uma das funções de ativação de uso mais frequente. Além de preservar a capacidade de aprendizado para problemas não lineares, em virtude da aplicação de uma transformação não linear nos *inputs*, a ReLU apresenta comportamentos similares às funções lineares, o que torna o cálculo dos gradientes mais simples e rápido. Além disso, não apresenta um limite superior para sua saída, o que contribui para que os gradientes se mantenham consistentes. Em contrapartida, quando o resultado da ativação se torna nulo, por exemplo pelo recebimento de *inputs* negativos, a unidade associada tornam-se incapaz de contribuir para o aprendizado do modelo (GOODFELLOW; BENGIO; COURVILLE, 2016).

Para contornar essa limitação, foram desenvolvidas variações da ReLU, como as abordagens *leak* ReLU e PReLU, que podem ser representadas por $max\{\alpha z, z\}$. Essas abordagens permitem que um *output* não nulo seja gerado quando z é negativo, o que viabiliza que a unidade se mantenha ativa e continue contribuindo para o processo de aprendizado. Ao mesmo tempo, controlam esse *output* para que ele não seja excessivamente negativo, o que poderia impactar a convergência do gradiente. Enquanto para a *leak* ReLU α é um hiperparâmetro definido para o modelo; para a PReLU, torna-se um novo parâmetro do próprio modelo, aprendido ao longo do treinamento (GÉRON, 2022).

Além das abordagens anteriores, outras funções de ativação aplicáveis para as camadas intermediárias são a logística e a tangente hiperbólica, embora tenham se tornado menos frequentes após a difusão da ReLU e suas variantes. Também denominada *sigmoid*, a função logística é expressa por $\sigma(z) = 1/(1 + \exp(-z))$, apresentando intervalo definido entre 0 e 1. A função tangente hiperbólica, por sua vez, pode ser descrita como $\tanh(z) = 2\sigma(2z) - 1$, possuindo intervalo entre -1 e 1. Embora sejam contínuas e diferenciáveis, o que seria positivo para o treinamento, essas funções acabam saturando tanto para os valores positivos quanto negativos de z, o que torna o aprendizado baseado em gradientes mais difícil de ser alcançado (GÉRON, 2022). Por conta desse comportamento, passaram a ser menos aplicadas diretamente como ativação nas camadas intermediárias em *feedforward neural networks*. Para outras abordagens de *deep learning*, como as *recurrent neural networks*, há outros requisitos e implementações que tornam o uso dessas funções de ativação mais adequado.

Para as unidades das camadas de *outputs* finais do modelo, a escolha da função de ativação está atrelada ao tipo de problema explorado. Em particular no presente trabalho, as abordagens de interesse são as classificações multi-classes ou multirrótulos de imagens. Em cenários de classificações multi-classes, em que a predição estará associada a somente uma das classes possíveis, aplica-se, usualmente, a função de ativação *softmax*. Para isso,

sendo $\mathbf{h}$ os *outputs* gerados pela última camada oculta, é realizada, inicialmente, uma camada de predições lineares, que pode ser representada por $\mathbf{z} = \mathbf{W}^T\mathbf{h} + \mathbf{b}$, em que $\mathbf{W}$ e $\mathbf{b}$ correspondem, respectivamente, à matriz de pesos e ao vetor de termos *bias* dessa camada. Cada um dos z_i *outputs* estão associados a probabilidade de cada classe i, mas não estão normalizados entre 0 e 1. Para realizar essa normalização, aplica-se a *softmax*, como representado pela expressão (GOODFELLOW; BENGIO; COURVILLE, 2016):

$$softmax(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \tag{2.3}$$

Para cada classe i, essa função de ativação gerará como predição a probabilidade dessa classe quando comparada às demais, de tal forma que a soma dessas probabilidades, considerando todas as j classes, será 1.

Já nos casos de classificações multirrótulos, em que a predição pode estar associada a mais de uma classe ao mesmo tempo, aplica-se, usualmente, a *sigmoide*. Assim como na abordagem anterior, é realizada, inicialmente, uma camada de predições lineares, que pode ser representada por $\mathbf{z} = \mathbf{W}^T\mathbf{h} + \mathbf{b}$. Após isso, aplica-se a *sigmoide* para cada um dos z_i *outputs*, como representado por $\sigma(z_i) = 1/(1 + \exp(-z_i))$. Assim, cada um desses *outputs* retornará um valor entre 0 e 1, representando a probabilidade da predição ser associada a cada classe i (GOODFELLOW; BENGIO; COURVILLE, 2016). Essa abordagem contempla, também, os casos de classificação binária. Para eles, haverá somente um *output* z e a probabilidade $\sigma(z)$ estimada será da classe positiva, podendo-se calcular $1 - \sigma(z)$ para encontrar a probabilidade da classe negativa (GÉRON, 2022).

2.2.5 Funções de custo

Como descrito para os métodos de otimização em 2.2.2, a função de custo, representada por $L(\mathbf{W})$, proporciona a medida de comparação entre os *outputs* gerados e os *targets* esperados, definindo o valor a ser minimizado em cada passo de atualização dos pesos do modelo. Nos problemas envolvendo classificações de imagens, as principais funções de custo aplicadas são adaptações do conceito de *cross-entropy*, dentre as quais a *categorical cross-entropy* para classificações multi-classes e a *binary cross-entropy* para as classificações binárias e multirrótulos. Originada do campo da Teoria da Informação, a *cross-entropy* é expressa por $H(p, q) = -\sum_x p(x) \log q(x)$ e mensura o quanto uma distribuição estimada de probabilidades $q(x)$ é similar à distribuição de probabilidades real dos *targets* $p(x)$, penalizando os casos que possuem um valor elevado para $p(x)$, mas não apresentaram uma estimativa associada elevada para $q(x)$ (GÉRON, 2022).

Aplicando esse conceito para um *input* i que pode ser classificado entre K classes, sendo $\hat{\mathbf{p}}^{(i)}$ e $\mathbf{y}^{(i)}$ os vetores com as probabilidades estimadas e as classificações reais de cada classe k, a *cross-entropy* desse *input* pode ser representada por:

$$H(\mathbf{y}^{(i)}, \hat{\mathbf{p}}^{(i)}) = -\sum_{k=1}^{K} {y_k}^{(i)} \log \hat{p_k}^{(i)} \tag{2.4}$$

Nos problemas multi-classe, em que as classes são mutuamente exclusivas, $\mathbf{y}^{(i)}$ será um vetor *one-hot*, com valor 1 para a classe válida e 0 para as demais. Nesse caso, sendo k a classe válida, a expressão 2.4 pode ser simplificada para $H(\mathbf{y}^{(i)}, \hat{\mathbf{p}}^{(i)}) = -\log \hat{p_k}^{(i)}$, abordagem denominada, também, de *categorical cross-entropy*. Já nos problemas de classificação binária, pode-se tanto manter a aplicação da expressão 2.4, quando as classes positiva e negativa estiverem representadas em um vetor *one-hot*, quanto ajustar a expressão para $H(y^{(i)}, \hat{p}^{(i)}) = -[y^{(i)} \log \hat{p}^{(i)} + (1 - y^{(i)}) \log(1 - \hat{p}^{(i)})]$, abordagem denominada *binary cross-entropy*. Por fim, nos problemas multirrótulos, pode-se tanto manter a expressão 2.4, sendo $\mathbf{y}^{(i)}$, nesse caso, um vetor composto por 1 em cada classe válida e 0 nas demais posições, quanto avaliar cada classe individualmente por meio da *binary cross-entropy*, somando, em seguida, os resultados obtidos (GÉRON, 2022).

É possível, também, generalizar a expressão 2.4 para todas as m' instâncias do *minibatch*, o que permite obter a seguinte função de custo, dados os $\mathbf{W}$ pesos do modelo:

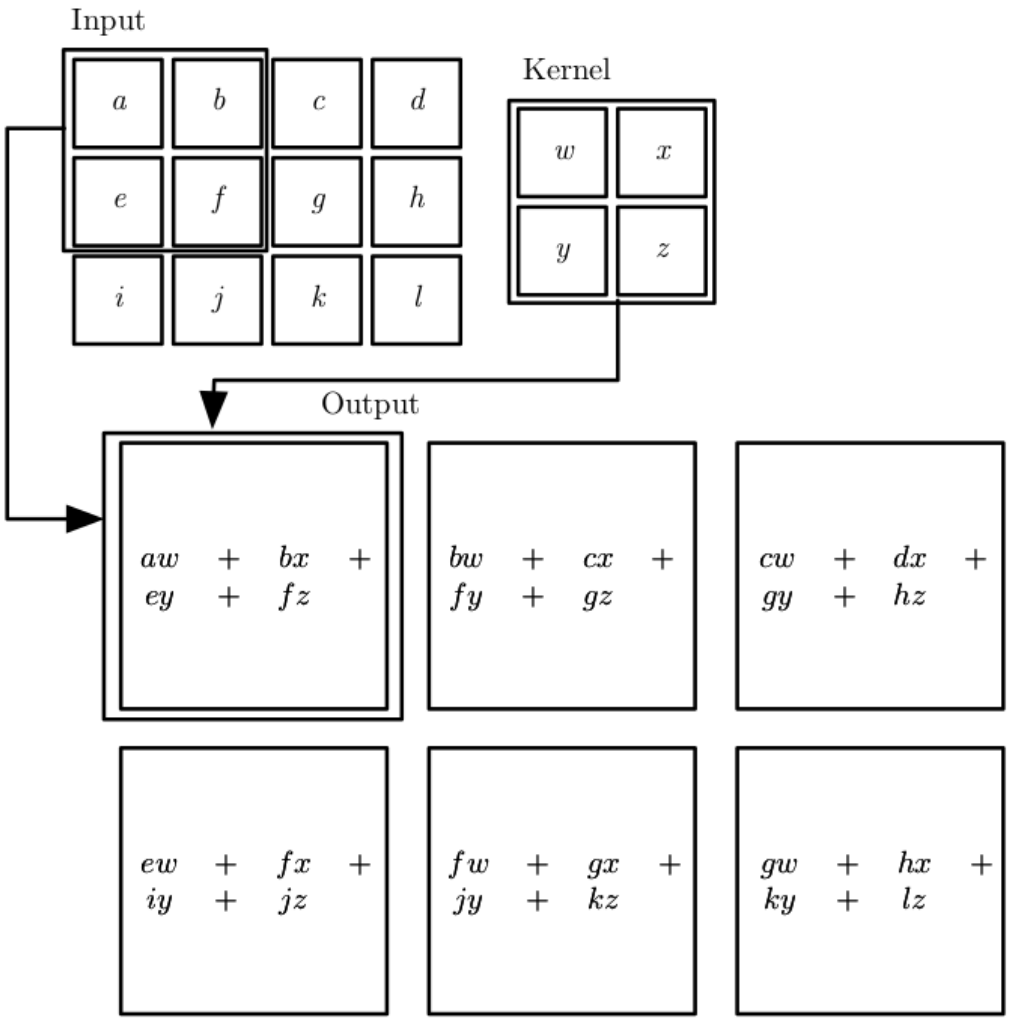
$$L(\mathbf{W}) = -\sum_{i=1}^{m'} \sum_{k=1}^{K} {y_k}^{(i)} \log \hat{p_k}^{(i)} \tag{2.5}$$

Como descrito nas seções anteriores, essa função de custo pode ser integrada ao processo de *backpropagation* para determinar o gradiente relacionado a cada um dos pesos, habilitando a aplicação dos algoritmos de otimização e a atualização desses pesos ao longo do processo de treinamento.

2.3 *Convolutional Neural Networks*

Como descrito por Goodfellow, Bengio e Courville (2016), as CNNs são definidas como *neural networks* que aplicam a operação de convolução nas suas camadas, sendo especializadas para processar dados que, como imagens, possuem estrutura topológica em *grid*. Enquanto nas *neural networks* com camadas totalmente conectadas os padrões são aprendidos globalmente a partir dos *inputs*, com os *outputs* de cada camada sendo associados a todos os *inputs* da camada anterior, pelas camadas de convolução, viabiliza-se o aprendizado de padrões locais. Para isso, elas apresentam um comportamento similar ao da Figura 1, apresentada na página seguinte. Por meio de um filtro, denominado *kernel*, que possui, por exemplo, altura f_h e largura f_w, percorre-se a camada de *input*, aplicando esse filtro ao longo dessa camada de tal forma que cada *output* gerado seja o produto dos pesos do *kernel* pelos *inputs* da região em que ele está sendo aplicado. Realizado esse processo, denomina-se como *feature map* a camada formada pelo conjunto de *outputs* da aplicação do *kernel*.

Figura 1 – Exemplo do processo de convolução



Fonte: Goodfellow, Bengio e Courville (2016)

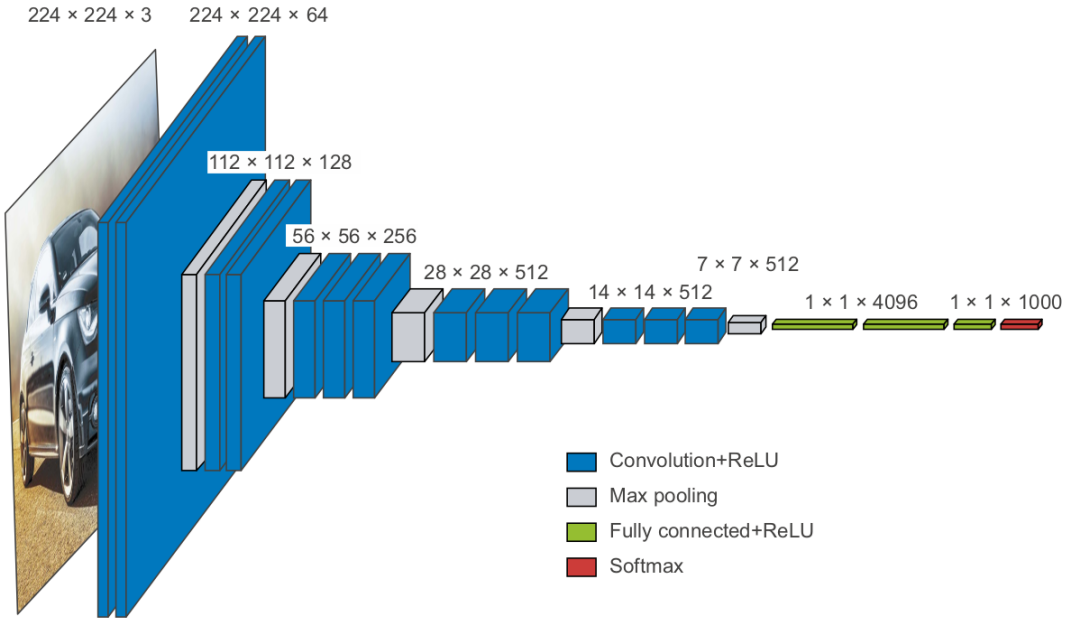
Na figura anterior, foi exemplificado um cenário em que os *inputs* iniciais eram formados a partir de uma camada bidimensional. Em imagens, no entanto, é comum que esses *inputs* sejam tridimensionais, associados, por exemplo, às cores do sistema RGB. Nesse caso, a diferença é que, inicialmente, o *kernel* é aplicado individualmente a cada uma das camadas de cores, denominadas *channels*. Em seguida, os resultados correspondentes são somados para gerar um único valor na camada de *outputs*, assim como ocorreria no cenário bidimensional (GÉRON, 2022). De forma similar, é comum que a camada de convolução seja formadas por múltiplos filtros. Nesse contexto, cada filtro aplicará o mesmo processo descrito anteriormente e gerará os respectivos *outputs*, de tal forma que, após a aplicação de todos os filtros, essa camada de convolução resultará em uma camada tridimensional, com a profundidade correspondendo à quantidade de filtros aplicados. Em seguida, essa camada de *outputs* pode ser enviada como *input* para uma nova camada convolucional, com o processo ocorrendo de forma análoga à anterior, com os novos *inputs* sendo tratados da mesma forma como descrito para os *channels* (GÉRON, 2022).

Para as camadas de convolução, dois conceitos adicionais relevantes são denominados *stride* e *padding*. O *stride* é um parâmetro que define qual será o deslocamento realizado pelo *kernel* à medida em que percorre os *inputs*. Na Figura 1, por exemplo, é possível avaliar que os deslocamentos foram de uma unidade. Menciona-se que, não necessariamente, os deslocamentos horizontal e vertical precisam receber o mesmo valor. É possível, por exemplo, que o *stride* horizontal seja de duas unidades, enquanto o vertical de somente uma. Já o *padding* é um parâmetro que controla os efeitos de borda, permitindo definir se os *outputs* apresentarão, ou não, as mesmas dimensões dos *inputs* recebidos (CHOLLET, 2021). Partindo da Figura 1, nota-se um caso com *padding* ausente. Nela, os cálculos foram restritos à região de *inputs*. Como consequência, as dimensões foram reduzidas de 3x4 nos *inputs* para 2x3 nos *outputs*. Caso o parâmetro *padding* fosse aplicado, seria como se as bordas dos *inputs* fossem preenchidas com valores nulos, viabilizando a aplicação do *kernel* nessas regiões, de tal forma que as dimensões do *output* seriam, também, 3x4.

Além das convoluções, um segundo tipo de camada usualmente presente nas CNNs são as *poolings*. Embora continuem extraindo características locais por meio de janelas que percorrem os *inputs*, elas fazem isso pela cálculo de uma função de agregação nessa região avaliada, retornando como *output*, por exemplo, o valor máximo (*max pooling*) ou médio (*average pooling*) da região. Com essas camadas, torna-se possível diminuir o custo computacional do modelo e a quantidade de parâmetros analisados (GÉRON, 2022).

Pela Figura 2, associada à VGG16, é possível observar um exemplo de arquitetura contemplando as camadas mencionadas.

Figura 2 – Arquitetura - VGG16



Fonte: Chollet (2021)

Nela, nota-se uma configuração modular e hierárquica. Partindo dos *inputs*, a rede

é formada por blocos de duas ou três camadas convolucionais e uma camada de *max pooling* que se sucedem. Observa-se, também, que esses blocos se distribuem formando uma estrutura similar a uma pirâmide, com o número de filtros aumentando à medida em que a profundidade das camadas se torna maior, ao mesmo tempo em que os tamanhos dos *feature maps* diminuem. Concluída essa sequência de blocos com camadas convolucionais e *poolings*, é definido o topo da rede, que é composto por duas camadas totalmente conectadas e pela camada final de predição. Apresentada para a VGG16, esse padrão de blocos de camadas que se organizam em uma estrutura similar a uma pirâmide é uma configuração típica para CNNs. Com esse padrão, à medida em que o tamanho dos *features maps* diminui e a profundidade das camadas aumenta, é ampliada de capacidade da rede em combinar as *features* das camadas precedentes e gerar níveis mais avançados de abstração (CHOLLET, 2021).

Embora o aumento da profundidade proporcione capacidade de abstração para a rede, à medida em que ela se torna mais profunda, amplia-se, também, o ruído presente nas camadas. Como consequência, quando esse ruído se torna excessivo, a rede perde capacidade de atualização correta dos gradientes e o processo de *backpropagation* deixa de funcionar. Para corrigir essa limitação, os modelos da família ResNet introduziram os blocos residuais. Esses blocos consistem em estruturas que armazenam os valores dos *inputs* recebidos e, após a aplicação das camadas associadas, adicionam, pelas conexões residuais, os *inputs* anteriormente armazenados aos *outputs* gerados, enviando o resultado dessa adição como *input* para as camadas seguintes. Nessa abordagem, as conexões residuais contribuem como um atalho de informação, evitando o impacto de blocos que poderiam gerar ruídos excessivos e, com isso, permitindo que a atualização dos gradientes possa acontecer adequadamente (CHOLLET, 2021). Nas etapas de modelagem do trabalho, as conexões residuais foram exploradas por meio da ResNet50, configuração que aplica os conceitos mencionados e contempla 50 camadas de convolução.

Além das abordagens anteriores, duas estratégias adicionais de modelagem, cujos aspectos conceituais são apresentados a seguir, são exploradas no trabalho visando contribuir para ganhos de performance nas classificações. A primeira delas corresponde à inclusão, ao longo da rede, de camadas de *batch normalization*, que conseguem proporcionar uma normalização adaptativa do conjunto de dados. Para que isso seja possível, durante o treinamento, essa camada realiza a normalização de seus *inputs* considerando os dados do *batch* vigente na etapa sendo treinada. Já na avaliação, a normalização é feita pela média móvel exponencial dos *batches* avaliados no treinamento. Além de tornar os *inputs* mais similares entre si, o que contribui para a capacidade de generalização do modelo, as camadas de *batch normalization*, assim como as conexões residuais, facilitam a propagação dos gradientes ao longo da rede, permitindo, também, que configurações mais profundas possam ser construídas (CHOLLET, 2021). Justamente por conta disso, é comum que essa camada componha as arquiteturas mais robustas. Isso ocorre, por exemplo, na ResNet50,

que combina conexões residuais e *batch normalization* em cada bloco.

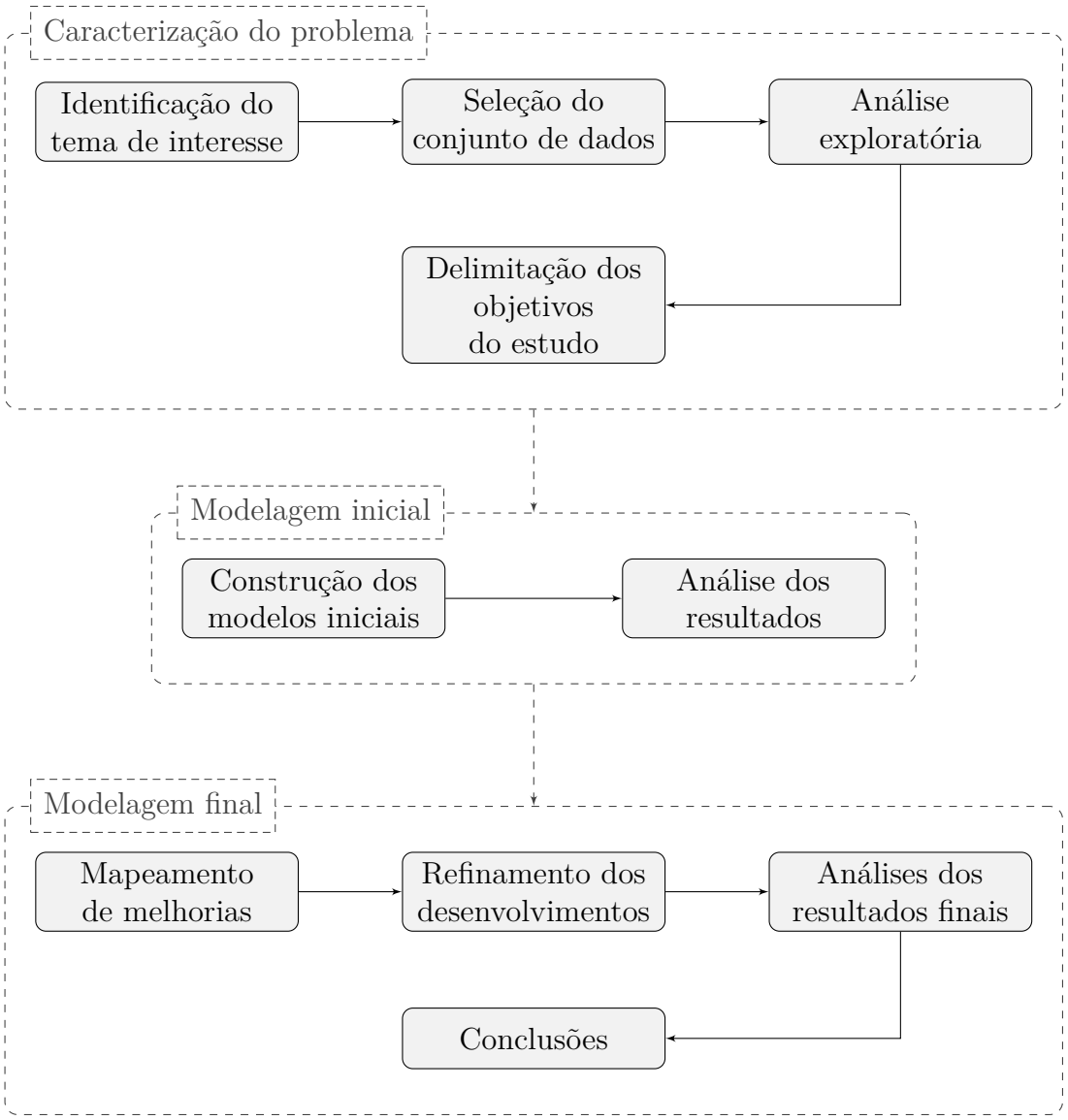
Por fim, a segunda estratégia adicional explorada correspondeu à inclusão de etapas de *dropout* anteriormente à camada de predição. Como apresentado por Géron (2022), essa abordagem consiste em uma técnica de regularização na qual, em cada etapa do treinamento, as unidades de uma camada apresentam probabilidade p de não estarem ativas e de não contribuírem para os *outputs* gerados. Como consequência dessa abordagem, o que se nota é que ela aprimora a capacidade de aprendizado da camada de uma forma geral. Isso ocorre pois, como em algum momento qualquer uma das unidades pode estar desativada, o modelo não pode depender de unidades específicas, o que contribuiu para que todas passem a apresentar uma relevância mais equilibrada na camada, auxiliando o modelo a se tornar mais generalizável.

Concluída a descrição dos fundamentos teóricos, aborda-se, no tópico seguinte, a metodologia do estudo, detalhando as atividades realizadas ao longo do trabalho.

3 METODOLOGIA

O diagrama a seguir ilustra a sequência de atividades realizadas no trabalho, que são descritas com maiores detalhes nos próximos tópicos.

Figura 3 – Sequência de atividades



Fonte: O autor (2023)

3.1 Caracterização do problema

Compondo a primeira etapa do estudo, a caracterização do problema abrangeu as atividades necessárias para direcionar e delimitar o escopo dos desenvolvimentos. Foi nesse momento em que se selecionou como ponto de partida para o trabalho o desafio *Planet: Understanding the Amazon from Space*, proposto na plataforma *Kaggle* em 2017

(GOLDENBERG *et al.*, 2017). Os dados desse desafio contemplam 81.148 imagens da Bacia Amazônica, coletadas entre janeiro de 2016 e fevereiro de 2017 por meio dos satélites *Flock* 2 da companhia *Planet*, sendo que 40.479 delas estão rotuladas e foram utilizadas para treinamento e a validação dos modelos. Individualmente, cada imagem possui dimensões de 256x256 *pixels*, cobrindo áreas de 89,72 hectares. No geral, a área abrangida pelo conjunto de todas as imagens é de 19.877,88 hectares (GOLDENBERG *et al.*, 2017).

Definido o conjunto de dados, a etapa seguinte do estudo consistiu em realizar a análise exploratória, que, como apresentado na seção de resultados, permitiu entender a distribuição dos rótulos e as características das imagens avaliadas. Foi a partir dessa análise que se definiram os objetivos específicos do trabalho, como detalhado na seção 1.2. Após isso, iniciaram-se as etapas de modelagem, seguindo as estratégias descritas a seguir.

3.2 Modelagens iniciais

Para as versões iniciais de modelagem, as imagens do conjunto de treinamento foram carregadas em memória de uma única vez. Para que isso fosse possível com os recursos computacionais disponíveis (processador 12th Gen Intel® Core™ i5-12500H, 16GB de RAM), foi necessário reduzir as dimensões durante o carregamento, transformando-as dos 256x256 *pixels* iniciais para 64x64. Após isso, as imagens foram normalizadas, tendo seus *pixels* convertidos para valores entre 0 e 1. Concluindo o pré-processamento, realizou-se a segregação dessas imagens nos conjuntos de treino e validação, optando-se por distribuí-las de forma aleatória, selecionando 80% dos dados para treino e 20% para validação.

Realizados os pré-processamentos anteriores, seguiu-se com a construção dos modelos, que foram configurados para predizerem todos rótulos das imagens de uma única vez. As alternativas de modelagem exploradas foram as seguintes:

- (1) Extração de *features* pela VGG16, predição por um classificador kNN;
- (2) *Transfer learning* pela VGG16;
- (3) *Transfer learning* pela ResNet50;
- (4) CNN com camadas definida por experimentação, sem *transfer learning*.

Menciona-se que, por ser um problema de classificação multirrótulo, a construção dos modelos seguiu os direcionamentos das seções 2.2.4 e 2.2.5, aplicando a *sigmoide* como função de ativação na camada de *outputs* e a *binary crossentropy* como função de custo. Além disso, utilizou-se a F_β *score* como métrica de avaliação. Recomendada nas orientações da competição, trata-se de uma medida que pondera precisão e revocação e permite controlar a influência do desbalanceamento das classes. Sendo a precisão representada por p e a revocação representada por r, a F_β pode ser calculada como:

$$F_\beta = (1 + \beta^2) \cdot \left(\frac{p \cdot r}{(\beta^2 \cdot p) + r} \right) \tag{3.1}$$

Todas as abordagens contemplaram, também, o processo de *early stopping* para controlar o *overfitting* dos modelos. Nas aplicações de *transfer learning*, além disso, foram testadas as inclusões de camadas de *batch normalization* e *dropout*. Os desempenhos de cada abordagem encontram-se detalhados em 4.2.

3.3 Modelagens finais

Após analisar os resultados iniciais, a etapa seguinte do estudo consistiu em mapear e implementar melhorias no pré-processamento dos dados e na configuração dos modelos.

Em relação ao pré-processamento, a primeira melhoria consistiu em modificar o modo de carregamento das imagens para ser realizado por meio de *batches*. Essa modificação permitiu que o consumo de memória no treinamento fosse otimizado e que as dimensões das imagens fossem aumentadas. Realizaram-se, nesses sentido, comparações dos modelos entre diferentes dimensões, variando-as desde 64x64 até 224x224. Em seguida, implementou-se a aumentação de dados para o conjunto de treinamento, variando o *zoom* das imagens, realizando *flips* horizontais e verticais e rotacionando-as para valores múltiplos de 90 graus.

Em relação às melhorias na configuração dos modelos, testou-se, inicialmente, ajustar o parâmetro *learning rate* para que ele fosse reduzido ao longo do treinamento, conforme as épocas não contribuíssem com melhorias na métrica de avaliação. Após isso, testou-se explorar dois modelos independentes, sendo um responsável por predizer as condições atmosféricas e o outro os elementos da superfície. Menciona-se que, como o modelo para a predição das condições atmosféricas passou a estar associado a um problema multi-classe, com rótulos mutuamente exclusivos, as funções de ativação da camada final e a função de custo foram ajustadas para a *softmax* e a *categorical cross-entropy*. Por fim, para o modelo dos elementos de superfície, testou-se variar o *threshold* de classificação das imagens entre cada um dos rótulos, que inicialmente estava definido em 0,5.

Considerando os experimentos descritos, os desempenhos proporcionados por cada abordagem encontram-se detalhados em 4.3.

4 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

Para detalhar os resultados do trabalho, inicialmente apresentam-se aspectos identificados na análise exploratória que contribuíram para direcionar o desenvolvimento dos modelos. Após isso, discutem-se os resultados obtidos com as modelagens, construídas seguindo as estratégias descritas nas seções 3.2 e 3.3.

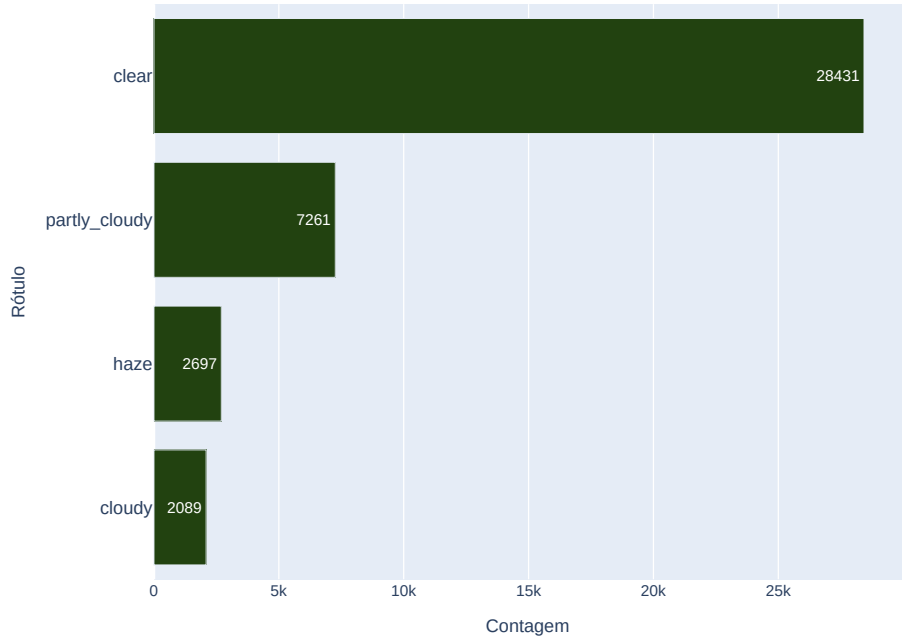
4.1 Análise exploratória

O objetivo inicial com a análise exploratória foi entender a distribuição dos rótulos das imagens. Como listado abaixo, foram encontradas 17 classes, divididas em três grupos:

- Condições atmosféricas: *clear*, *partly cloudy*, *cloudy* e *haze*.
- Elementos comuns de superfície: *primary*, *agriculture*, *water*, *habitation*, *road*, *cultivation* e *bare ground*.
- Elementos raros de superfície: *slash burn*, *selective logging*, *blooming*, *conventional mine*, *artisinal mine* e *blow down*.

Para os rótulos de condições atmosféricas, a contagem de ocorrências de cada classe no conjunto de treinamento pode ser avaliada no gráfico apresentado abaixo.

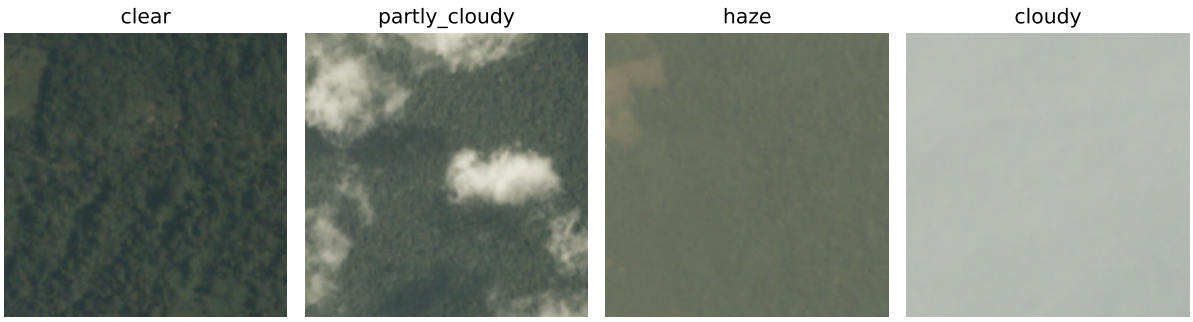
Figura 4 – Ocorrências dos rótulos de condições atmosféricas



Fonte: O autor (2023)

Por esse gráfico, nota-se que mais de 70% das imagens estão associadas ao rótulo *clear*, não apresentando nuvens ou neblinas. Avalia-se, ainda, que as imagens totalmente nubladas são as menos frequentes, ocorrendo em 5% dos casos. Para elas, os rótulos de superfície não são classificáveis. Isso pode ser observado, por exemplo, na imagem *cloudy* da Figura 5. Além disso, como pode ser avaliado também pela Figura 5, para as condições atmosféricas, cada imagem pode estar associada a somente uma das classes. Foi, justamente, com a validação desse comportamento que a construção de modelos segregados para condições atmosféricas e elementos de superfície foi mapeada como uma estratégia adequada para ser explorada no trabalho, como será detalhado em 4.3.

Figura 5 – Exemplos de imagens para rótulos de condições atmosféricas

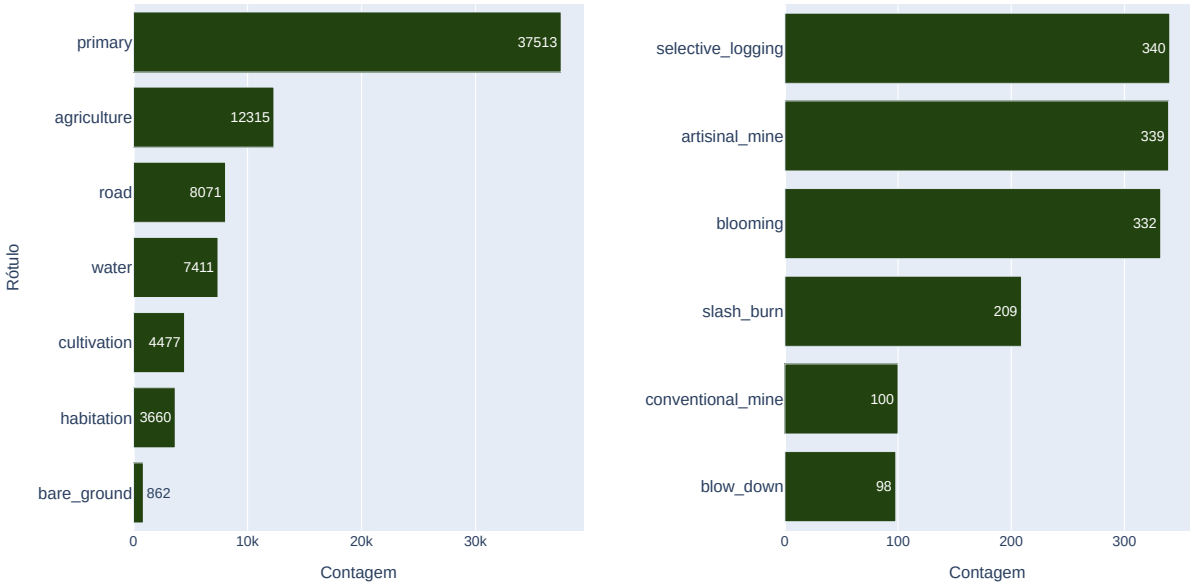


Fonte: Goldenberg *et al.* (2017)

Já para os elementos de superfície, as distribuições dos rótulos nos cenários de elementos comuns e raros podem ser observadas nos gráficos da Figura 6.

Figura 6 – Distribuição dos rótulos para os elementos de superfície

(a) Elementos comuns (b) Elementos raros



Fonte: O autor (2023)

Um primeiro aspecto observável pelo gráfico de elementos comuns é que o total

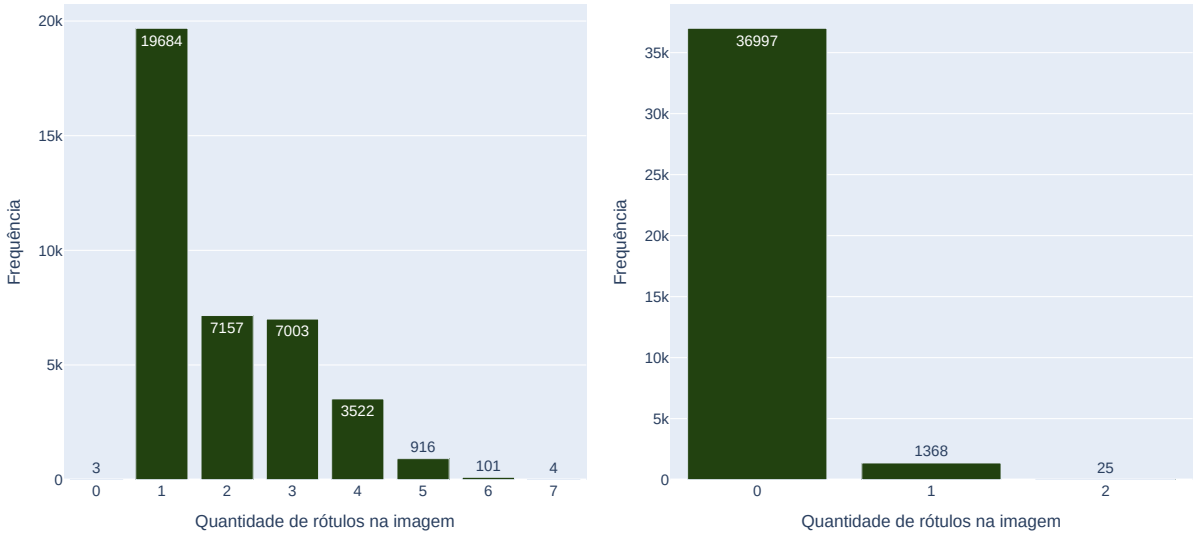
de classificações, somado em 74.309, é superior à quantidade de imagens do conjunto de treinamento, composto por 40.479 imagens. Isso evidencia que, diferentemente do ocorrido com as condições atmosféricas, uma mesma imagem pode abranger múltiplos rótulos de superfície. Pode-se avaliar, ainda, que o rótulo de vegetação primária é o mais comum, ocorrendo em 93% das imagens. Comparado à soma de todos os demais desse grupo, esse rótulo é cerca de 2,3 vezes mais presente. Isso sugere ser frequente na base de treinamento a ocorrência de imagens similares à *partly cloudy* da Figura 5, que possui somente vegetação primária como elemento de superfície. Esse desbalanceamento entre as classes justifica, inclusive, a escolha da F_β como métrica de avaliação, já que, com ela, consegue-se mensurar de forma equilibrada o resultado geral, sem que ele seja excessivamente influenciado pela classe mais frequente.

Já pelo gráfico de elementos raros, nota-se que, de fato, os rótulos desse grupo estão significativamente menos presentes nas imagens de treinamento do que os rótulos comuns. O rótulo *selective logging*, por exemplo, que é o rótulo raro de maior frequência e está associado à atividade de remoção de árvores selecionadas da vegetação primária, é cerca de 2,5 vezes menos frequente do que o rótulo *bare ground*, que é o de menor ocorrência para os elementos comuns e se associa ao cenário de desmatamento completo, com presença de solo exposto. Ao comparar a soma dos rótulos raros com a soma dos rótulos comuns, nota-se que, mesmo sem considerar a classe de vegetação primária, os rótulos raros estão 25 vezes menos presentes no conjunto de treinamento. Como consequência, como será apresentado nos resultados dos modelos, esses rótulos são os mais difíceis de serem preditos.

A Figura 7, por sua vez, ilustra a distribuição da quantidade de rótulos de superfície por imagem para os casos em que a condição atmosférica não é totalmente nublada.

Figura 7 – Histogramas da quantidade de rótulos por imagem

(a) Elementos comuns de superfície (b) Elementos raros de superfície

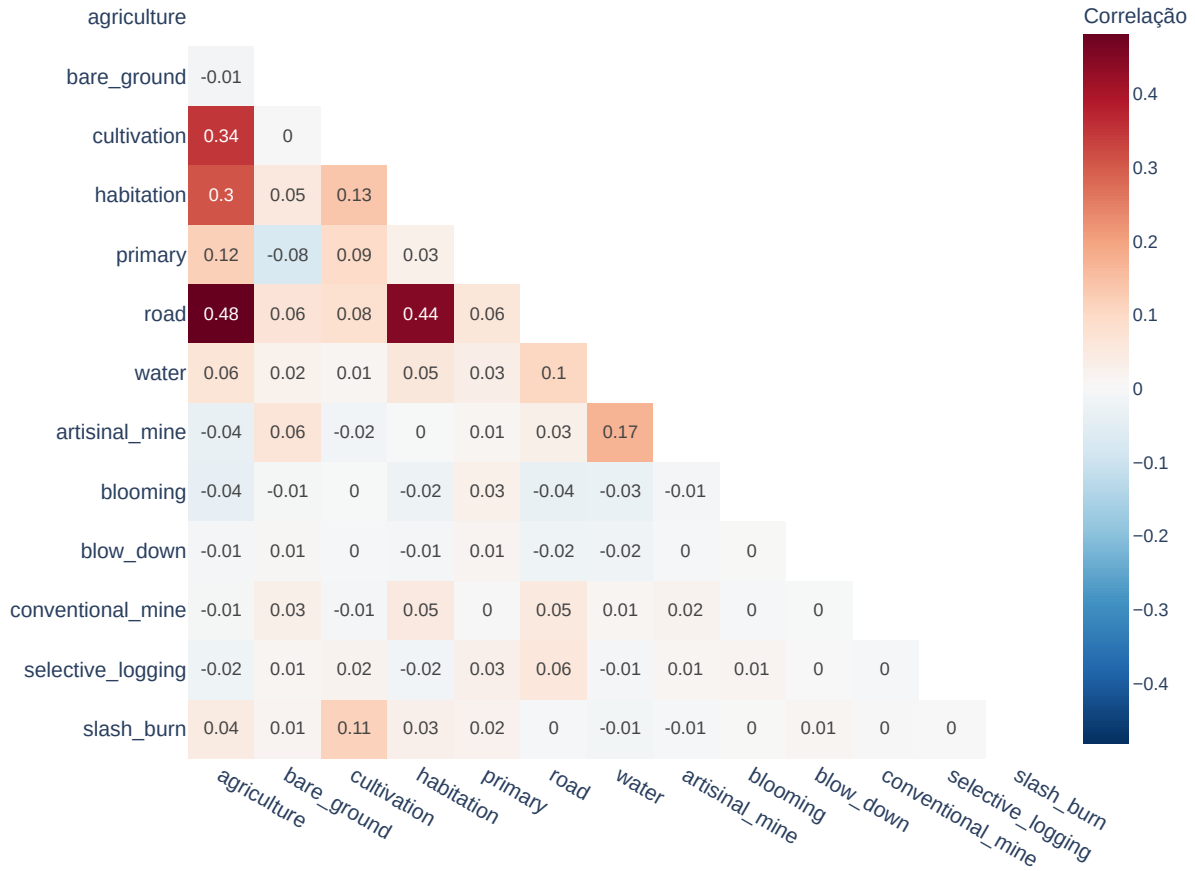


Fonte: O autor (2023)

Pode-se avaliar, nesses histogramas, que, para os elementos raros de superfície, somente 3,6% das imagens possuem algum rótulo e 0,06% apresentam mais de 1. Já para os elementos comuns, nota-se que somente 3 imagens não apresentam rótulos e que, embora o cenário com uma única classe seja o mais recorrente, cerca de 48% das imagens apresentam mais de uma classe, sendo frequentes cenários de 2, 3 ou 4 rótulos.

Já a figura seguinte ilustra a matriz de correlação entre os elementos de superfície, permitindo avaliar quais rótulos tendem a ocorrer de forma conjunta ou oposta entre si.

Figura 8 – Matriz de correlação entre os elementos de superfície



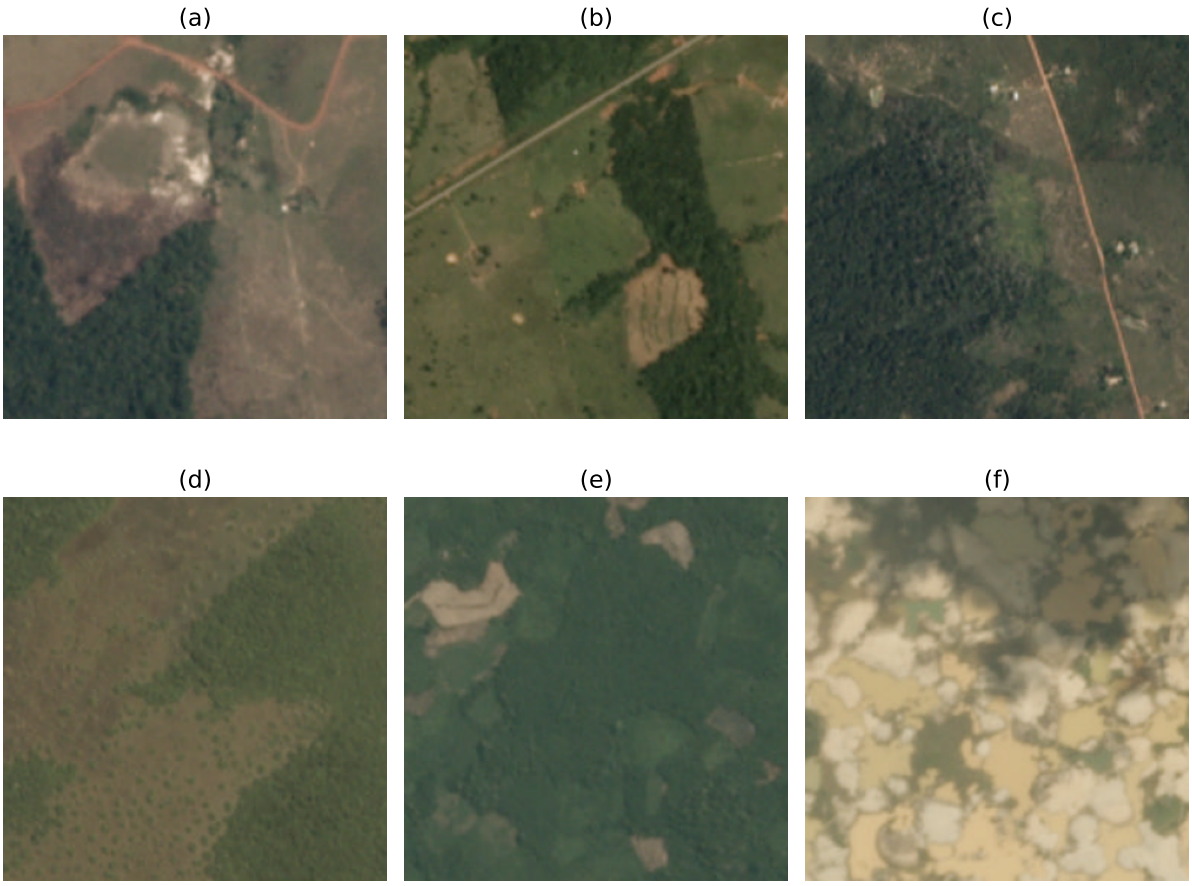
Fonte: O autor (2023)

Nessa matriz, nota-se que as maiores correlações são entre *agriculture* e *road*, com valor 0,48, e entre *habitation* e *road*, com valor 0,44. É interessante avaliar, também, que esses rótulos apresentam correlações altas com a classe *cultivation*, como pode ser verificado, em especial, para o rótulo *agriculture*, para o qual a correlação é de 0,34. Os valores elevados de correlação para esse conjunto de classes é coerente, já que, combinadas, elas descrevem contextos de ocupação humana nos locais das imagens analisadas. A matriz permite avaliar, também, que o menor valor de correlação, de −0,08, ocorre entre *primary* e *bare ground*, cenário que pode ser explicado pelo rótulo *bare ground* representar, como mencionado anteriormente, situações em que a vegetação primária é completamente

retirada, resultando em solo exposto. Para os elementos raros de superfície, duas correlações mostram-se particularmente interessantes. A primeira delas, com valor 0,11, é entre *slash burn* e *cultivation*, podendo ser explicada pelas atividades de corte e queimada, mesmo que ilegalmente, serem muitas vezes aplicadas próximas às regiões de cultivo para que a região de produção possa ser expandida. A segunda correlação que se destaca, com valor 0,17, é aquela entre *artisinal mine* e *water*, que pode ser explicada pelo termo *artisinal mine* estar associado às atividades de garimpo, que tendem a ocorrer próximas aos leitos d'água.

Por fim, os exemplos de imagens apresentadas seguintes permitem visualizar alguns dos cenários descritos nas análises das correlações.

Figura 9 – Exemplos de imagens para rótulos de superfície



Fonte: Goldenberg *et al.* (2017)

Nesse sentido, nas imagens (b) e (c), podem ser observadas regiões de agricultura, cultivo, habitação e estrada. Na imagem (d), é possível avaliar um exemplo de solo exposto, próximo a uma região de vegetação com degradação progressiva. Já nas imagens (a) e (e), podem ser visualizadas ocorrências de queimadas próximas às regiões de vegetação primária e cultivo. Por fim, na imagem (f), é possível avaliar uma região de garimpo.

4.2 Resultados dos modelos iniciais

Apresentada a análise exploratória, detalham-se, a seguir, os resultados dos modelos iniciais. Nessa etapa, como mencionado em 3.2, as imagens de treinamento foram carregadas em memória de uma vez, reduzidas dos 256x256 *pixels* iniciais para 64x64. Do total de 40.479 imagens, selecionaram-se, aleatoriamente, 32.383 para compor o conjunto de treino e 8.096 o de validação. As estratégias de modelagem foram:

- (1) Extração de *features* pela VGG16, predição por um classificador kNN;
- (2) *Transfer learning* pela VGG16;
- (3) *Transfer learning* pela ResNet50;
- (4) CNN com camadas definida por experimentação, sem *transfer learning*.

Nas abordagens (1) a (3), utilizaram-se os pesos da *ImageNet* para iniciar o treino dos modelos. Além disso, para (1), as *features* foram extraídas da camada *block5_pool* da VGG16, a última antes do topo da rede. Já para (2) e (3), nos primeiros modelos, congelaram-se as camadas anteriores ao último bloco, realizando o treinamento nesse bloco e no topo, que, nas duas abordagens, foi formado pelas seguintes camadas: *global average pooling*, densa com 512 unidades (ativação *relu*) e densa com 17 unidades (camada final de predição). Para o modelo (5), foi construída a arquitetura representada na Tabela 1.

Tabela 1 – Arquitetura da CNN definida por experimentação

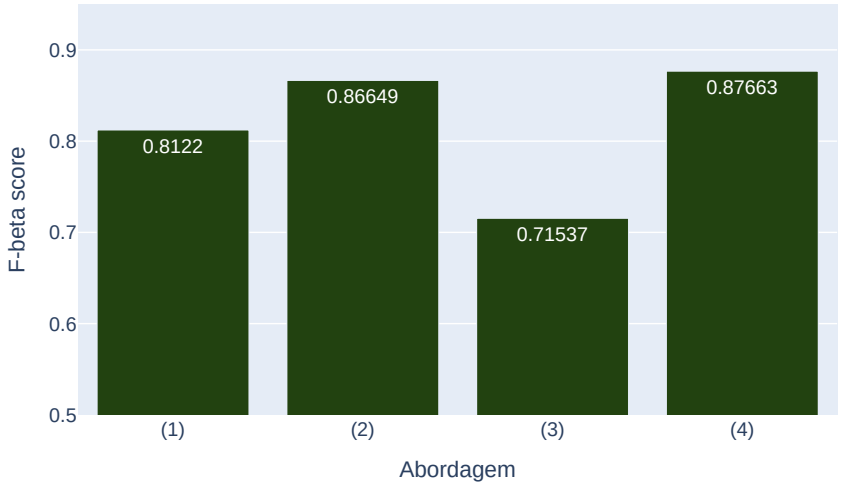
layer	*shape*
conv2d	(None, 64, 64, 32)
conv2d	(None, 64, 64, 32)
max_pooling2d	(None, 32, 32, 32)
conv2d	(None, 32, 32, 64)
conv2d	(None, 32, 32, 64)
max_pooling2d	(None, 16, 16, 64)
global_average_pooling2d	(None, 64)
dense	(None, 128)
dense	(None, 17)

Fonte: O autor (2023)

No gráfico 10, ilustram-se os resultados das abordagens (1) a (4) nesse cenário inicial. Nessa figura, nota-se que a CNN definida por experimentação apresentou o melhor resultado no conjunto de validação, seguida da VGG16, com 1,2% de diferença na performance entre elas. Um aspecto interessante, também, foi a diferença na quantidade de épocas necessárias para que o melhor resultado fosse alcançado nessas abordagens. Enquanto para a VGG16 o aproveitamento dos pesos pré-treinados da rede permitiu que a melhor métrica fosse

alcançada após somente 5 épocas; para a CNN experimental, foram necessárias 31 épocas. Como consequência, embora cada época da VGG16 fosse cerca de 4 vezes mais demorada do que a da CNN experimental, ao avaliar o cenário geral, a VGG16 exigiu um tempo total de treinamento 58% menor do que aquele da CNN experimental.

Figura 10 – Resultados dos modelos para o cenário 1 das modelagens iniciais



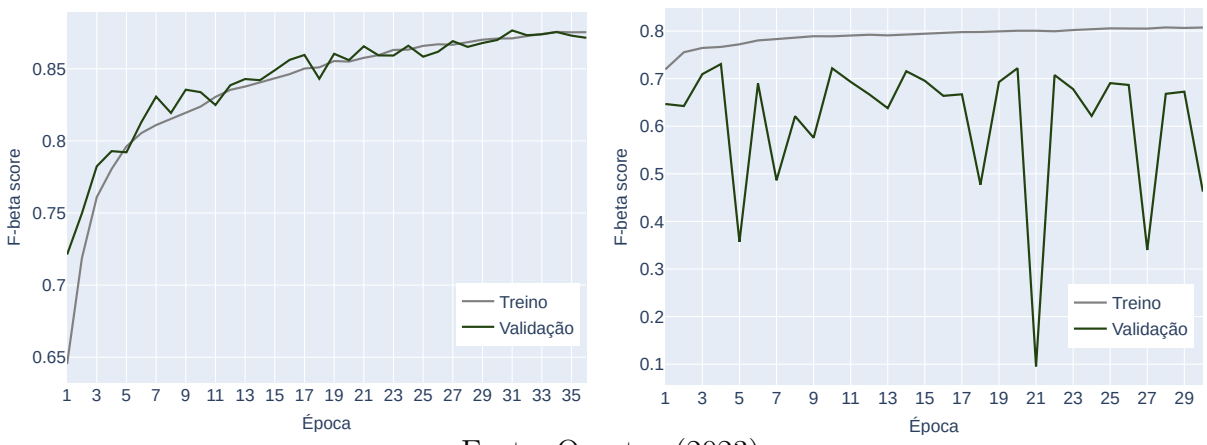
Fonte: O autor (2023)

Pela Figura 10 é possível avaliar, ainda, que a ResNet50 foi a abordagem de pior resultado nesse cenário inicial. Esse comportamento pode ser justificado por ela ser uma rede de maior complexidade e pelo uso dos seus parâmetros pré-treinados ser diretamente influenciado pelas dimensões iniciais da imagem. Nesse sentido, como nesse cenário foi necessário reduzir substancialmente as dimensões, isso impactou diretamente a performance dessa arquitetura e contribuiu para que outras modelagens mais simples, em especial a abordagem (4), apresentassem resultados superiores. Complementando essa análise, os gráficos seguintes apresentam as curvas de aprendizado das abordagens (3) e (4).

Figura 11 – Curvas de aprendizado dos modelos

(a) Abordagem 4 - CNN experimental

(b) Abordagem 3 - ResNet50

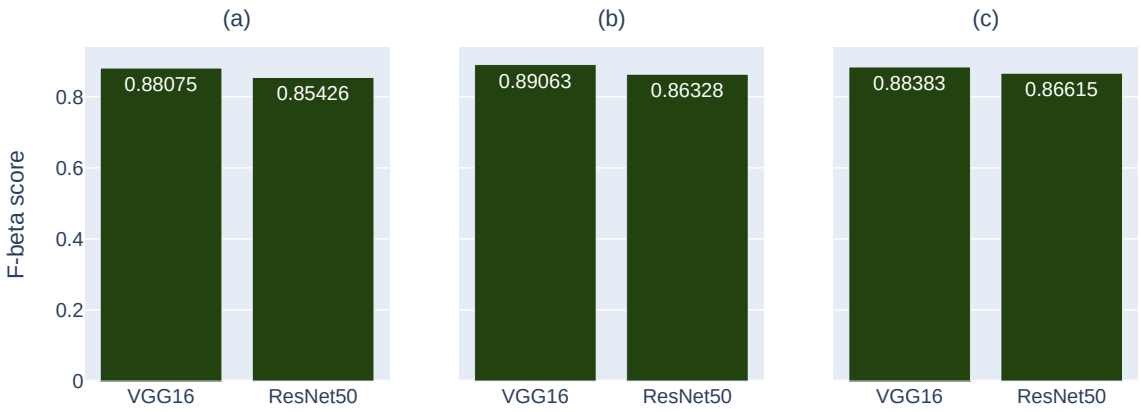


Fonte: O autor (2023)

Por meio deles, pode-se notar que enquanto as curvas de treino e validação progridem de forma conjunta para a CNN experimental, reduzindo gradualmente os erros do modelo; para a ResNet, os resultados na validação são piores do que os de treino em todas as épocas, existindo um descolamento entre as curvas que se expande ao longo do treinamento. Observa-se, ainda, que o resultado no conjunto de validação é irregular, como uma oscilação significativa na performance entre as épocas. Em conjunto, esses aspectos reforçam o entendimento de que o uso de *transfer learning* pela ResNet50 não foi uma abordagem adequada para classificar as imagens nesse primeiro cenário, caracterizado pela redução das dimensões dos *inputs* e pelo congelamento dos parâmetros anteriores ao último bloco.

A Figura 12, por sua vez, apresenta os resultados da aplicação da VGG16 e da ResNet50 para três novas configurações. No cenário (a), os pesos da *ImageNet* continuaram sendo utilizados no início do treinamento, mas descongelaram-se todos os blocos da rede, permitindo que todos os parâmetros fossem atualizados desde a primeira época. Na configuração (b), incluiu-se à modificação do cenário (a) uma camada de *batch normalization* para a leitura dos *inputs*. Por fim, além das mudanças da configuração (b), o cenário (c) adicionou no topo da rede, antes da predição, uma camada de *dropout* com valor 0,15.

Figura 12 – Resultados dos modelos iniciais para as configurações avaliadas



Fonte: O autor (2023)

Por esses gráficos, avalia-se, inicialmente, que o resultado da ResNet50 melhorou de forma relevante já no cenário (a), avançando em 20% em relação à Figura 10. Menciona-se, também, que esse ganho de performance está em linha com as discussões apresentadas sobre o problema de compatibilidade desse modelo à redução nas dimensões das imagens, já que, com a modificação do cenário (a), os parâmetros dos blocos mais próximos aos *inputs* iniciais tiveram liberdade para serem retreinados, ajustando-se às imagens fornecidas. A inclusão das camadas de *batch normalization* e *dropout* contribuíram, também, para ganhos de performance no uso da ResNet. No entanto, como se observa na Figura 10, esses ganhos foram menos representativos do que o proporcionado pelo descongelamento dos blocos.

Já para a VGG16, observa-se que a configuração (b) foi capaz de proporcionar os melhores resultados, com um ganho de 2,8% em relação à Figura 10. Essa maior

performance, em contrapartida, exigiu um tempo total treinamento aproximadamente 4 vezes maior. De maneira mais detalhada, enquanto no contexto da Figura 10 foram necessárias 8 épocas, com cerca de 4 minutos cada, para completar o treinamento; para a configuração (b), foram necessárias 15 épocas de, aproximadamente, 8 minutos. Menciona-se, também, que esse aumento no tempo esteve associado, sobretudo, ao descongelamento dos blocos, já que esse comportamento ocorreu de forma similar no cenário (a), para o qual também foram necessárias 15 épocas de 8 minutos durante o treinamento. É interessante avaliar, ainda, que o modelo composto pela VGG16 na configuração (b) foi aquele com o melhor resultado geral no contexto dos modelos iniciais, proporcionando um ganho de 1,5% na performance em relação ao resultado da CNN experimental. Na Tabela 2, são detalhados os resultados desse melhor modelo para cada rótulo.

Tabela 2 – Detalhamento do resultado por rótulo

Grupo	Rótulo	Casos (treino)	Casos (validação)	F_β
Condições atmosféricas	clear	22.703	5.728	0,970
	cloudy	1.694	395	0,774
	haze	2.164	533	0,648
	partly_cloudy	5.821	1.440	0,903
Elementos de superfície comuns	agriculture	9.874	2.441	0,852
	bare_ground	687	175	0,007
	cultivation	3.613	864	0,449
	habitation	2.917	743	0,672
	primary	30.006	7.507	0,989
	road	6.503	1.568	0,793
	water	5.952	1.459	0,650
Elementos de superfície raros	artisinal_mine	269	70	0,469
	blooming	268	64	-
	blow_down	73	25	-
	conventional_mine	78	22	-
	selective_logging	279	61	-
	slash_burn	162	47	-

Fonte: O autor (2023)

Nela, observa-se, inicialmente, que os elementos raros de superfície possuem valores ausentes de F_β. Isso indica que, pela baixa ocorrência desses rótulos no treino, com exceção da classe *artisinal mine*, o modelo não aprendeu a identificar os rótulos desse grupo. E, mesmo para a classe *artisinal mine*, a performance foi baixa, como se nota pelo valor 0,469 de F_β apresentado na tabela. Aprofundando na análise dos erros dessa classe, a Tabela 3 detalha a contagem dos rótulos reais e preditos para as imagens que possuíam o elemento *artisinal mine*. Por ela, avalia-se que o modelo identificou cerca de 43% das ocorrências desse rótulo. Observa-se, além disso, que os valores preditos das classes *agriculture*, *habitation* e *road* são maiores do que os reais. Isso indica que, influenciado pelo desbalanceamento das classes e por aspectos de similaridade visual, nas situações em que o modelo não identificou

o rótulo *artisinal mine*, ele interpretou a presença desse elemento como se fosse associada a essas outras três classes, mais frequentes no conjunto de treino.

Tabela 3 – Classes reais e preditas para imagens com rótulo *artisinal mine*

	artisinal mine	agriculture	habitation	road
Real	70	11	2	21
Predito	30	20	36	43

Fonte: O autor (2023)

Nas imagens da Figura 13, é possível avaliar três desses casos de inversão da classificação do rótulo *artisinal mine*. Nas imagem (a), o modelo confundiu a região de garimpo com os elementos *agriculture*, *road* e *habitation*. Já nas imagens (b) e (c), as inversões ocorreram com os elementos *road* e *habitation*.

Figura 13 – Exemplos de imagens *artisinal mine* com classificação invertida

Fonte: Goldenberg *et al.* (2017)

Quanto aos elementos comuns de superfície, na Tabela 2, observa-se que os rótulos *bare ground* e *cultivation* foram os mais difíceis de serem preditos. Para eles, assim como realizado com o *artisinal mine* na análise dos erros, avaliaram-se as contagens reais e preditas para as imagens em que essas classes estavam presentes. Em relação ao *bare ground*, nota-se, na Tabela 4, que o modelo conseguiu identificar apenas 1 das ocorrências do rótulo, confundindo-o, principalmente, com a classe *agriculture*. Esse comportamento pode ser justificado pelo desbalanceamento entre essas classes no treino e pela existência de aspectos de similaridade visual, que ocorrem, sobretudo, quando a presença de solo exposto não é tão intensa e se assemelha a regiões em estágios iniciais de produção agrícola.

Tabela 4 – Classes reais e preditas para imagens com rótulo *bare ground*

	bare ground	agriculture	habitation	primary	road
Real	175	45	35	142	57
Predito	1	141	53	172	77

Fonte: O autor (2023)

Já para o elemento *cultivation*, verifica-se, pela Tabela 5, que o modelo encontrou cerca de 42% das ocorrências do rótulo e que os cenários de classificação errada aconteceram com o modelo confundindo esse elemento com *agriculture*, *habitation* e *road*. Para esses erros, além de aspectos de similaridade visual, o que pode ter contribuído são, como discutido na análise exploratória, as maiores correlações existentes entre o rótulo *cultivation* e as classes *agriculture*, *habitation* e *road*. Como mencionado, nas imagens que descrevem regiões de ocupação humana, é comum que esses rótulos ocorram de forma conjunta. Isso pode ter gerado um viés no conjunto de treino, provocando o maior erro na etapa de validação.

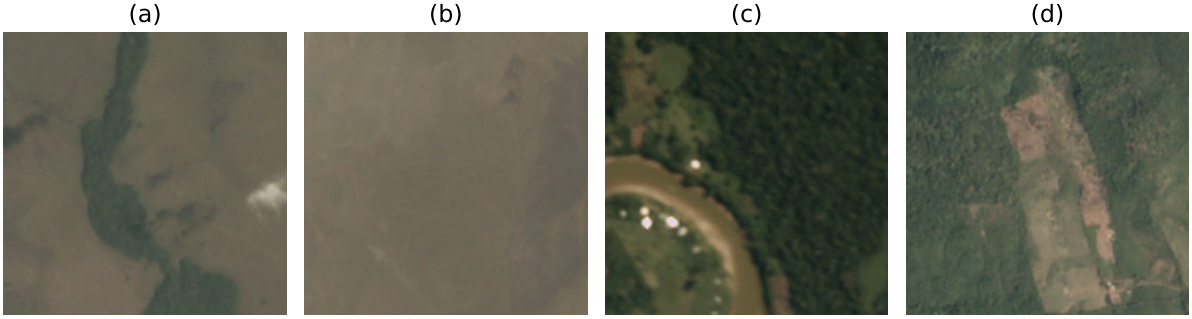
Tabela 5 – Classes reais e preditas para imagens com rótulo *cultivation*

	cultivation	agriculture	habitation	road
Real	864	656	182	238
Predito	364	720	237	293

Fonte: O autor (2023)

Na Figura 14, ilustram-se imagens em que o modelo não foi capaz de identificar os elementos *bare ground* e *cultivation*. Para (a) e (b), que seriam associadas ao rótulo *bare ground*, o modelo apresentou o comportamento mencionado de classificar essas imagens como *agriculture*, possivelmente por confundi-las com regiões em estágio iniciais de produção agrícola. Já para (c) e (d), que seriam associadas ao rótulo *cultivation*, possivelmente pelo viés do treinamento associado às correlações, interpretou que os rótulos *agriculture*, *habitation*, *road* estariam presentes nas duas imagens, embora em (c) não sejam visíveis os elementos *road* e *agriculture* e em (d) não seja visível o elemento *road*.

Figura 14 – Exemplos de imagens *bare ground* e *cultivation* com classificação invertida



Fonte: Goldenberg *et al.* (2017)

Por fim, em relação aos rótulos de condições atmosféricas, a Tabela 2 permite avaliar que o maior erro esteve associado à classe *haze*. Para as imagens com esse rótulo, as classificações erradas ocorreram, principalmente, com o modelo interpretando que elas pertenceriam à classe *clear*, como pode ser observado pela Tabela 6. Além disso, avaliou-se que essas inversões de classificação estiveram associadas, no geral, a imagens com pouca variação nos elementos de superfície, sobretudo em casos em que havia somente vegetação

primária presente. Nesses casos, o modelo apresentou dificuldade em diferenciar que a tonalidade um pouco mais opaca existente na imagem seria da presença de neblina, e não do próprio elemento de superfície.

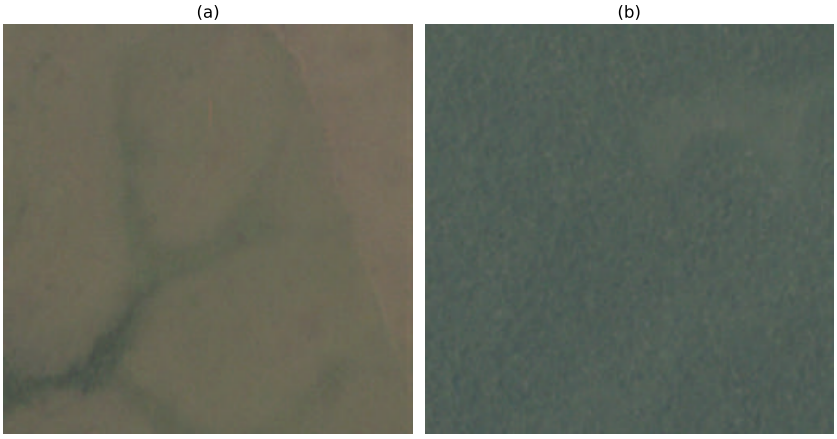
Tabela 6 – Classes reais e preditas para imagens com rótulo *haze*

	clear	cloudy	haze	partly cloudy
Real	0	0	533	0
Predito	145	16	333	24

Fonte: O autor (2023)

As imagens da Figura 15 ilustram duas das ocorrências em que o modelo confundiu os rótulos *haze* e *clear*. Esses dois casos exemplificam o cenário mencionado de imagens com pouca diversidade nos elementos de superfície, nas quais o modelo não conseguiu diferenciar se a opacidade existente seria do próprio elemento ou da presença de neblina.

Figura 15 – Exemplos de imagens *haze* classificadas como *clear*



Fonte: Goldenberg *et al.* (2017)

4.3 Resultados dos modelos finais

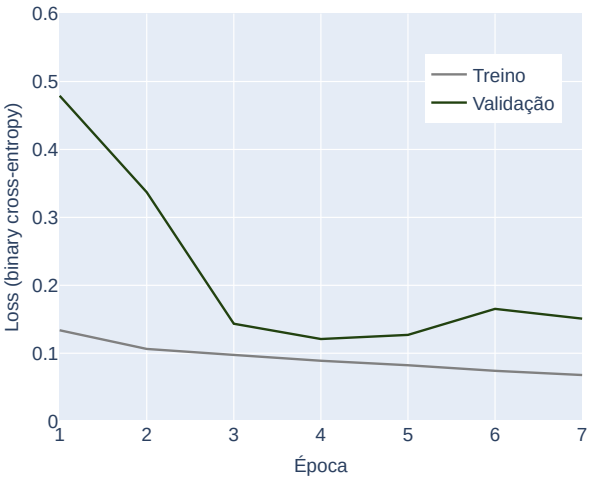
Concluída a análise de resultados dos modelos iniciais, a seguir, discutem-se como as melhorias implementadas nos modelos finais aprimoraram a capacidade de classificação das imagens. Nesse estágio da pesquisa, a primeira mudança foi ajustar a configuração do parâmetro *learning rate* para que ele pudesse ser atualizado ao longo das épocas, reduzindo seu valor à medida em que os pesos do modelo se aproximassem da configuração ideal, visando facilitar a convergência da otimização. Com mais detalhes, enquanto esse parâmetro foi definido como 10^{-3} para todas as épocas dos modelos iniciais; nos modelos finais, foi ajustado para ser reduzido em 10 vezes sempre que uma época não apresentasse ganhos de performance, sendo limitado a um valor de 10^{-7}.

Considerando essa mudança, o resultado pela VGG16 melhorou, mas não de forma substancial, passando de 0,891 no melhor modelo inicial para 0,894 na abordagem com

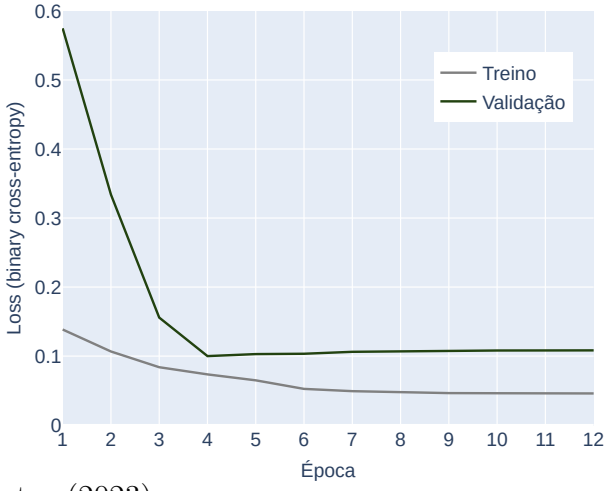
learning rate variável. Já para a ResNet50, a variação foi mais significativa, com o resultado avançando de 0,866 para 0,895. Pelas curvas de erro da ResNet na Figura 16, nota-se que o cenário com *learning rate* variável é capaz, na validação, tanto de atingir um valor menor de erro do que o cenário com *learning rate* fixa quanto de, alcançado esse valor, manter-se mais estável ao longo das épocas, com uma evolução da aprendizagem mais similar nos conjuntos de treino e validação. Já no contexto de *learning rate* fixa, nota-se um comportamento irregular nas épocas, com a curva de validação oscilando após atingir o valor mínimo e, no geral, se afastando da curva de treino, o que sugere a ocorrência de um problema de *overfitting* nesse cenário, evitado na abordagem com *learning rate* variável.

Figura 16 – Curvas de aprendizado contendo, ou não, variação no *learning rate*

(a) Caso com *learning rate* fixa



(b) Caso com *learning rate* variável



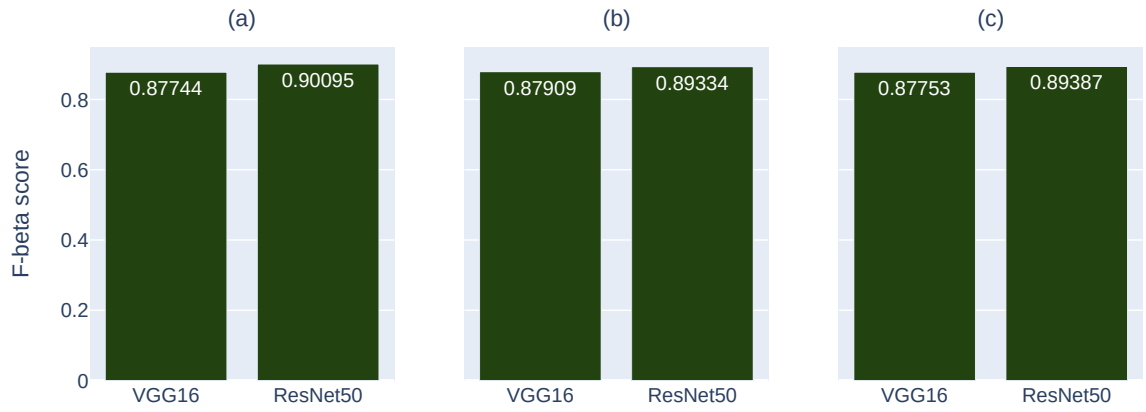
Fonte: O autor (2023)

Concluídos os experimentos com a *learning rate*, a etapa seguinte foi explorar a aumentação dos dados de treino, como representado na Figura 17 para os cenários (a), (b) e (c). No cenário (a), testaram-se rotações das imagens em múltiplos de 90º e espelhamentos horizontais e verticais. No cenário (b), além das aumentações de (a), testaram-se *zooms* de forma aleatória, focando ou afastando as imagens em um intervalo entre 85% e 115% em relação às imagens originais. Já no cenário (c), aplicou-se *zoom* somente por aproximação, com um intervalo entre 85% e 100% da imagem original, o que foi realizado para evitar o surgimento de ruídos nas bordas, efeito que acaba acontecendo no cenário com afastamento.

Pela Figura 15, observa-se que a ResNet50 no cenário (a) foi o modelo com o melhor resultado, sendo o primeiro do trabalho a superar o patamar de 0,9 de performance. É interessante avaliar, também, que a aplicação do *zoom*, nos dois cenários explorados, acabou não contribuindo para um aumento representativo na diversidade dos dados de treino e os resultados nesses cenários acabaram sendo inferiores, para a ResNet50, e similares, para a VGG16, em relação àqueles do experimento (a). Nota-se, ainda, que, com a maior complexidade dos dados de treino proporcionada pela aumentação, comparado aos experimentos anteriores, em que existia uma proximidade maior de performance entre

VGG16 e ResNet50, nesses novos cenários, a superioridade nos resultados da ResNet50 em relação à VGG16 passou a ser mais evidente, o que ocorreu em conjunto com um menor tempo de treinamento para a ResNet50 em relação à VGG16. Enquanto cada época da VGG16 levou cerca de 8 minutos para ser concluída; para a ResNet50, exigiu cerca de 5 minutos. Em conjunto, a melhor performance e o menor tempo de treinamento motivaram que a ResNet50 fosse a configuração selecionada para os experimentos seguintes realizados.

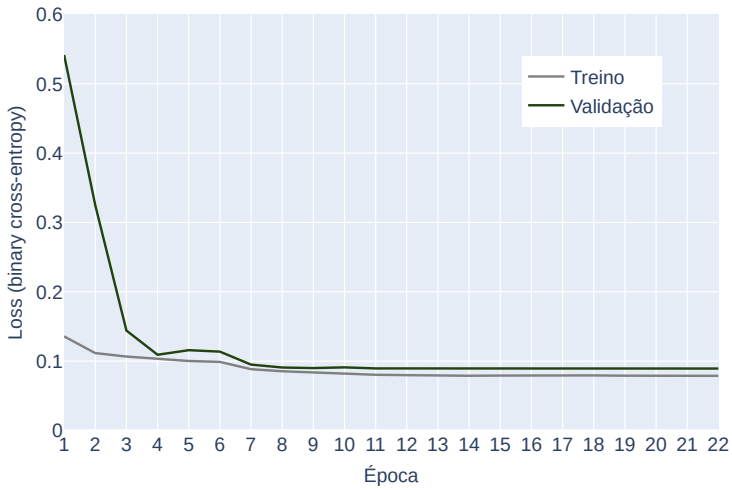
Figura 17 – Resultados dos modelos iniciais para as configurações avaliadas



Fonte: O autor (2023)

Por fim, menciona-se que, com a maior diversidade nos dados de treino, a diferença de performance nos conjuntos de treino e validação se tornou menor para os experimentos com aumentação do que nos anteriores. Isso pode ser observado, por exemplo, pela Figura 18, que ilustra a curva de aprendizado para a ResNet50 no cenário (a). Comparando-a às curvas de aprendizado apresentadas na Figura 16, nota-se que, para o cenário com aumentação, o distanciamento entre as curvas de treino e validação é significativamente menor ao longo das épocas, evidenciando que, nesse cenário, o processo de treinamento se tornou mais representativo dos resultados a serem medidos no contexto de validação.

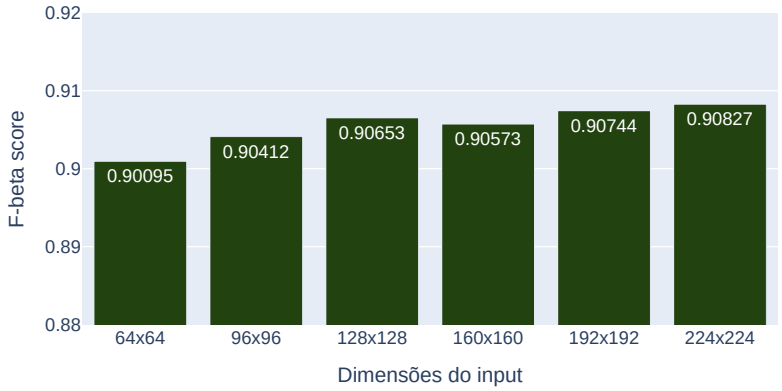
Figura 18 – Curva de aprendizado para a ResNet50 com aumentação dos dados



Fonte: O autor (2023)

Analisados os experimentos com aumentação de dados, apresentam-se, a seguir, os resultados obtidos pela variação das dimensões das imagens. Partindo do melhor modelo anterior com a ResNet50, as dimensões foram ampliadas para até 224x224 *pixels*. Pela Figura 19, observa-se que as dimensões relacionaram-se diretamente com a performance da classificação. Esse comportamento pode ser justificado pelo *inputs* fornecidos para o modelo tornarem-se mais próximos aos *pixels* das imagens iniciais à medida em que as dimensões aumentam, reduzindo a perda de informação na modelagem. Além disso, com as maiores dimensões, a estrutura do modelo tornou-se mais próxima àquela em que os pesos foram originalmente treinados na *ImageNet*, contribuindo para que esses pesos fossem mais compatíveis com o contexto da classificação desde o início do treinamento.

Figura 19 – Resultados da ResNet50 com a variação nas dimensões das imagens

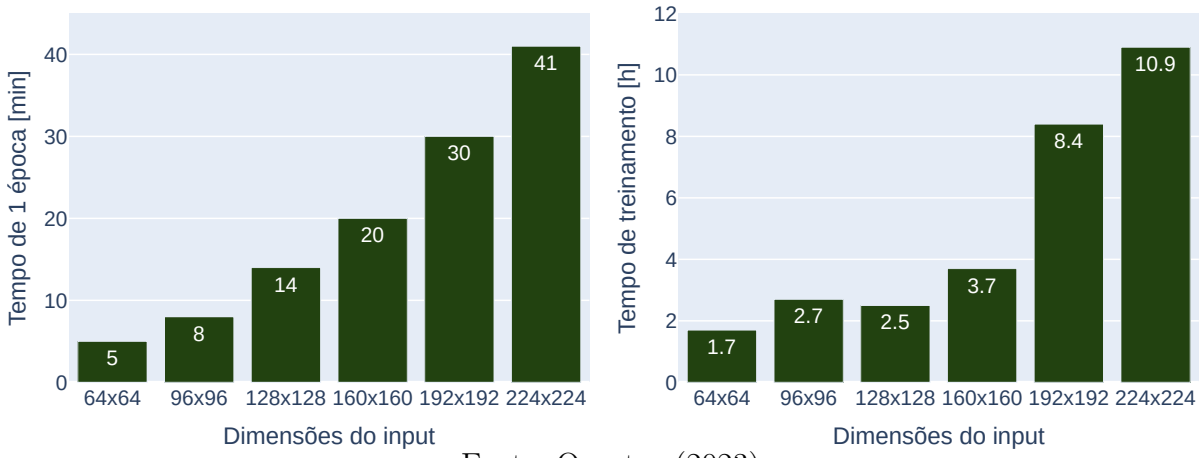


Fonte: O autor (2023)

Por outro lado, embora o modelo com dimensões 224x224 tenha proporcionado o melhor resultado, ele exigiu um tempo de treinamento substancialmente maior. Como apresentado na Figura 20, para esse modelo, cada época demorou cerca de 8 vezes mais que o cenário 64x64 e, para o treinamento completo, o tempo exigido foi 6,4 vezes maior.

Figura 20 – Tempos de treinamento em função das dimensões dos *inputs*

(a) Tempo para uma época (b) Tempo total

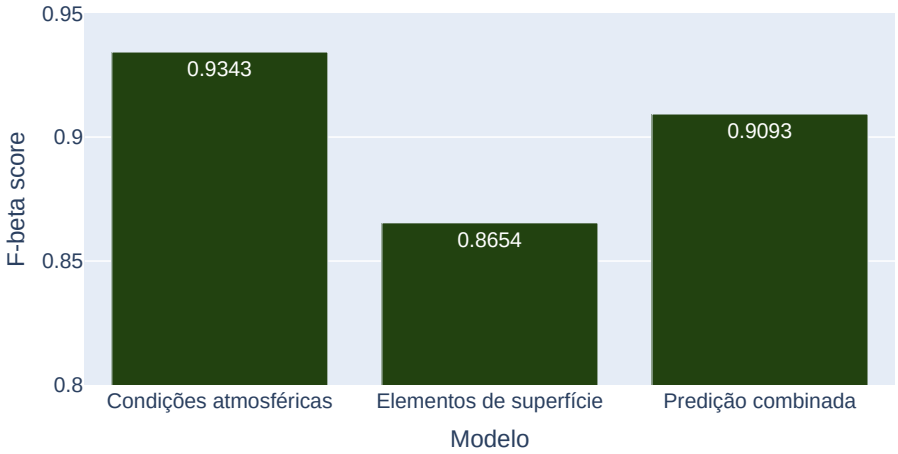


Fonte: O autor (2023)

É interessante observar na Figura 20, também, o comportamento do modelo 128x128. Para ele, foram necessárias 11 épocas até a conclusão do treinamento, enquanto o modelo inicial 64x64 exigiu 22 épocas. Como consequência, embora o tempo de cada época do cenário 128x128 tenha sido, aproximadamente, 2,8 vezes maior do que o inicial, o tempo total de treino entre eles variou somente em 0,8 horas. Além disso, a performance desse modelo foi somente 0,2% pior do que aquela do cenário 224x224. Considerando esse contexto, avalia-se que, caso o presente trabalho estivesse associado a uma aplicação real em que o processo de treinamento fosse frequente e o tempo de treinamento fosse uma restrição, a dimensão ideal, embora não proporcionasse o melhor resultado, seria a 128x128.

Analisados os experimentos anteriores, apresentam-se, a seguir, os resultados do contexto de modelos segregados para as condições atmosféricas e para os elementos de superfície. Nessa nova abordagem, os modelos foram construídos partindo da configuração da ResNet50 com *inputs* de dimensões 224x224. Para o modelo de elementos de superfície, os únicos ajustes foram na quantidade de *outputs* da camada final de predição, alterada para as 13 possíveis classes, e na seleção dos dados de treino/validação, dos quais foram removidas as imagens totalmente nubladas. Já para o modelo de condições atmosféricas, como mencionado em 3.3, as funções de ativação na camada de predição e de custo foram ajustadas para o contexto multi-classe. Por fim, para avaliação do cenário combinado no conjunto de validação, realizou-se, inicialmente, a predição das condições atmosféricas. Em seguida, para as imagens não preditas como totalmente nubladas, realizou-se a classificação dos elementos de superfície. Os resultados dos modelos individuais, assim como da predição combinada, encontram-se apresentados a seguir, na Figura 21.

Figura 21 – Resultados da abordagem por modelos segregados



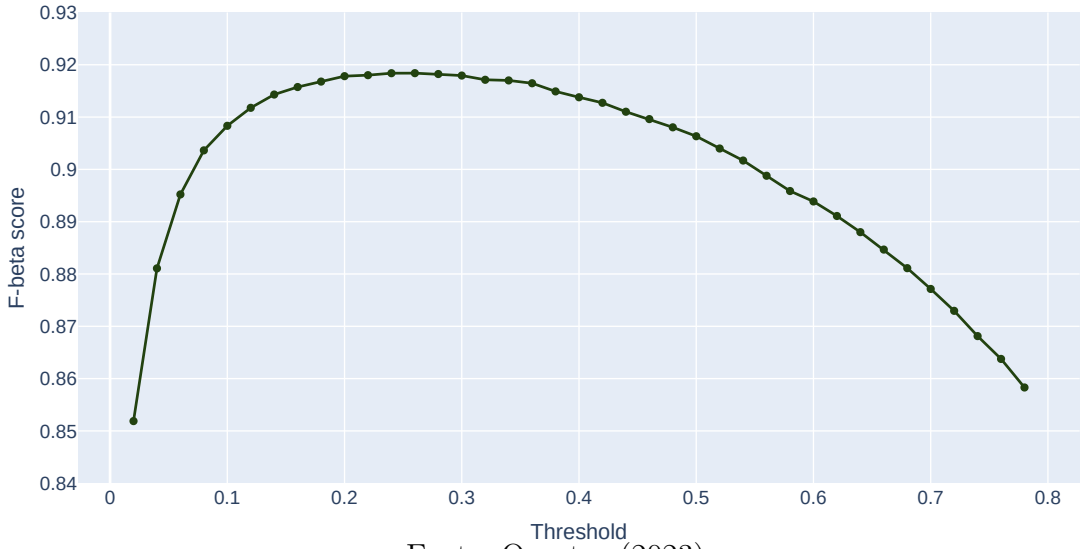
Fonte: O autor (2023)

A partir desse gráfico, ao avaliar o resultado da predição combinada, é interessante observar que, mesmo com modelos específicos para condições atmosféricas e elementos de superfície, a performance aumentou somente 0,1% em relação ao melhor cenário de predição de todas as classes de uma vez. Isso evidencia que, mesmo avaliando todas as

classes, o modelo único foi capaz de lidar com cada uma delas de forma independente. Dado esses resultados, caso os experimentos fossem finalizados nessa etapa, a modelagem única seria considerada a melhor, já que, além de ser mais simples de ser construída por abrange somente um modelo, apresentaria resultados similares à exploração de modelos segregados. Por outro lado, com a construção de modelos segregados, foi possível explorar a variação do *threshold* de classificação dos rótulos de superfície, mantendo o modelo de condições atmosféricas prevendo somente um rótulo por imagem. Como será apresentado a seguir, com essa melhoria, o cenário de modelos segregados passou a proporcionar ganhos representativos de performance, consolidando essa abordagem como a melhor do trabalho.

A Figura 22 ilustra, justamente, a variação, em função do *threshold*, da performance da classificação dos elementos de superfície para as imagens preditas como não sendo totalmente nubladas no modelo de condições atmosféricas. Por esse gráfico, nota-se que o melhor valor de *threshold* correspondeu a 0,26, associado a um F_β próximo a 0,92 para os rótulos de superfície nesse conjunto de imagens. Partindo desse cenário e avaliando o novo resultado das predições combinadas, a nova performance mensurada foi de 0,918, representado um ganho de 1% em relação ao melhor cenário sem a variação do *threshold* e de 3% em relação ao melhor modelo avaliado na etapa dos resultados iniciais.

Figura 22 – Resultados variando o *threshold* de classificação dos elementos de superfície



Fonte: O autor (2023)

Contribuindo para entender o significado desse ganho de 3%, a Tabela 7 compara, por rótulo, os resultados dos melhores modelos inicial e final. Por ela, é interessante avaliar que, mesmo com performances baixas em virtude da pequena representatividade dos elementos raros de superfície no conjunto de treino, o modelo aprendeu a classificar parte das ocorrências desses rótulos, diferentemente do modelo inicial, que sempre previa esses elementos como ausentes. Nota-se que o único rótulo em que esse comportamento persistiu foi o *slash burn*. Para essa classe, por aspectos de semelhança visual e de viés das ocorrências conjuntas dos rótulos no treinamento, o modelo continuou confundindo

as aparições do *slash burn* com se fossem associadas às classes *agriculture* e *cultivation*. Já para o *artisinal mine*, que o modelo conseguia classificar parcialmente, mas confundia com os elementos *agriculture*, *habitation* e *road*, a performance melhorou em cerca de 65%, ganho que pode ser justificado, sobretudo, pela redução dos casos em que o *artisinal mine* era confundido com a classe *habitation*. Antes, como apresentado na Tabela 3, embora existissem 2 casos reais do rótulo *habitation* para as imagens *artisinal mine*, o modelo previa 36 ocorrências. Após as melhorias, essas previsões foram reduzidas para 10 ocorrências.

Tabela 7 – Detalhamento do resultado por rótulo para o modelo final

Grupo	Rótulo	F_β inicial	F_β final	Variação
Condições atmosféricas	clear	0,970	0,971	0,001
	cloudy	0,774	0,833	0,059
	haze	0,648	0,645	-0,003
	partly_cloudy	0,903	0,925	0,022
Elementos de superfície comuns	agriculture	0,852	0,894	0,042
	bare_ground	0,007	0,311	0,304
	cultivation	0,449	0,648	0,199
	habitation	0,672	0,793	0,121
	primary	0,989	0,990	0,001
	road	0,793	0,870	0,077
	water	0,650	0,804	0,154
Elementos de superfície raros	artisinal_mine	0,469	0,771	0,302
	blooming	-	0,226	0,226
	blow_down	-	0,187	0,187
	conventional_mine	-	0,485	0,485
	selective_logging	-	0,396	0,396
	slash_burn	-	-	-

Fonte: O autor (2023)

Já em relação ao elementos comuns de superfície, os principais ganhos foram nas classes *bare ground* e *cultivation*. Assim como para os modelos iniciais, eles continuaram sendo os rótulos de maior erro, mas tiveram suas performances aprimoradas significativamente. Para o *bare ground*, enquanto antes o modelo havia conseguido identificar somente 1 das 175 ocorrências do rótulo no conjunto de validação, com o novo modelo, esse número aumentou para 50. Para esse caso, a melhoria foi proporcionada pela redução no *threshold*, diferentemente do que ocorreu com a classe *artisinal mine*, em que o novo modelo foi capaz de reduzir as inversões de rótulo. Assim, nas imagens *bare ground*, o modelo continuou interpretando que o elemento *agriculture* estaria presente, mas com o menor *threshold*, foi capaz de classificar, também, o rótulo correto *bare ground*. Para o rótulo *cultivation*, o comportamento foi similar. Antes, como mencionado nos resultados iniciais, o modelo havia conseguido classificar 42% das ocorrências do rótulo, confundindo-o com os elementos *agriculture*, *habitation* e *road*. Após as melhorias, esse número subiu para 69,6%, mas as classificações dos rótulos errados continuaram presente, indicando que a melhoria no

resultado foi proporcionada, assim como no caso do *bare ground*, pelo menor *threshold*.

Para os rótulos de condições atmosféricas, observa-se na Tabela 7 que, apesar das melhorias na modelagem, o rótulo *haze* continuou sendo aquele com a pior performance. Para essa classe, persistiu o cenário analisado nos modelos iniciais de que, em imagens com pouca variação nos elementos de superfície, o modelo apresenta dificuldade em diferenciar a tonalidade um pouco mais opaca presente na imagem como pertencente à neblina, e não ao próprio elemento de superfície. Já como cenário positivo para esse grupo de rótulos, avalia-se que os principais ganhos foram nas classes *cloudy* e *partly cloudy*, com melhorias de 7,6% e 2,4%. Para elas, a Tabela 8 apresenta a contagem dos rótulos reais e preditos nos cenários inicial e final. Em relação ao *cloudy*, nota-se que a melhoria do resultado foi proporcionada pela correção dos casos em que esse rótulo havia sido confundido, principalmente, com a classe *partly cloudy*. Já para o rótulo *partly cloudy*, as correções foram, sobretudo, nos rótulos que haviam sido interpretados como *clear*.

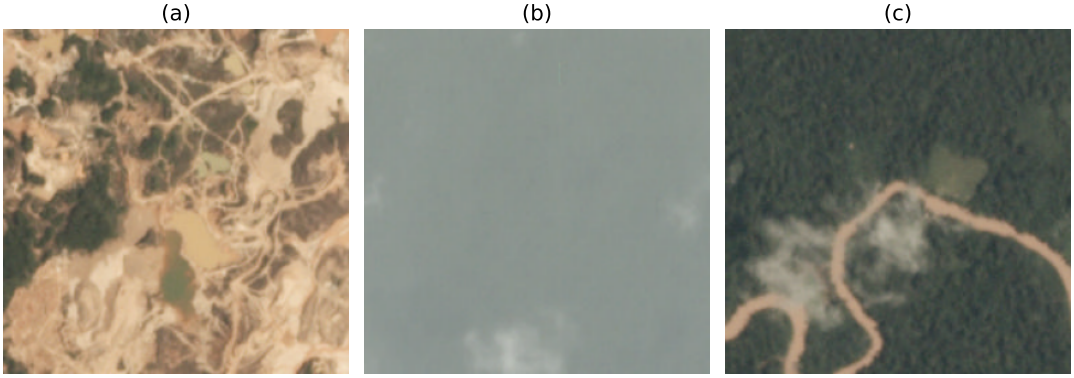
Tabela 8 – Comparativo das classificações de condições atmosféricas

Rótulo	Cenário	cloudy	partly cloudy	clear	haze
cloudy	Real	395	0	0	0
	Predito (inicial)	296	37	16	37
	Predito (final)	324	25	14	29
partly cloudy	Real	0	1440	0	0
	Predito (inicial)	5	1297	119	9
	Predito (final)	13	1339	65	6

Fonte: O autor (2023)

Concluindo a seção de resultados, a Figura 23 ilustra três imagens que, classificadas com inversão de rótulo no modelo inicial, foram interpretadas corretamente no modelo final. Nesse sentido, (a) ilustra um caso de *artisinal mine* que, anteriormente, era classificado como *habitation*; (b) um caso de *cloudy* que era classificado como *partly cloudy* e (c) um caso de *partly cloudy* que era classificado como *clear*.

Figura 23 – Exemplos de imagens com classificações corrigidas no modelo final



Fonte: O autor (2023)

5 CONCLUSÕES

Aplicando distintas técnicas de *deep learning*, o trabalho apresentado construiu modelos de visão computacional voltados à classificação de múltiplos rótulos presentes em imagens de satélite da região amazônica. Para isso, utilizou como base de dados um conjunto com cerca de 40 mil imagens disponibilizadas na plataforma *Kaggle* no ano de 2017. Consultadas essas imagens, as principais etapas do trabalho foram de análise exploratória e de desenvolvimentos dos modelos iniciais e finais.

Na análise exploratória, para os rótulos de condições atmosféricas, o estudo avaliou que cada imagem poderia estar associada a somente uma classe, o que motivou a construção de um modelo multi-classe específico para esses rótulos na etapa dos resultados finais. Já para os elementos de superfície, foi observado que, embora o cenário com um único rótulo fosse o mais comum, cerca de 48% das imagens apresentavam mais de uma classe, sendo comuns cenários com 2, 3 ou 4 rótulos. Nesse grupo de rótulos, notou-se, também, que existia uma variação significativa na representatividade entre as classes, com o rótulo *primary* sendo 2,3 vezes mais frequente do que a soma de todos os demais desse grupo. Para os elementos raros, notou-se que, de fato, a presença desses elementos nas imagens era esporádica, ocorrendo somente em 3,6% das imagens disponíveis. Como consequência, esse desbalanceamento trouxe desafios que o trabalho procurou superar, mas que acabaram refletidos na performance dos modelos desenvolvidos.

Nos modelos iniciais, após explorar quatro alternativas de modelagem, o trabalho concluiu que a melhor abordagem correspondeu ao caso em que VGG16 foi configurada com uma camada de *batch normalization* no início da rede e foi ajustada para que os pesos da *ImageNet*, em todas as camadas, possuíssem liberdade de retreino desde a primeira época. Nessa abordagem, a performance geral pela métrica F_β foi de 0,891, mas apresentou variação significativa entre os rótulos. Em particular, o trabalho observou que, pela baixa representatividade no conjunto de treino, o modelo foi incapaz de classificar elementos raros de superfície, prevendo, no geral, a ausência dos elementos desse grupo. Já para os elementos comuns, embora o modelo tenha apresentado performance elevada nos rótulos *agriculture* e *primary*, acabou, por efeitos de similaridade de aspectos visuais e viés do conjunto de treino, invertendo classificações e apresentando performance piores nos rótulos *bare ground* e *cultivation*. Por fim, para as condições atmosféricas, o modelo apresentou performance elevada na identificação das imagens *clear*, mas, nos casos com neblina e pouca variação de elementos de superfície, apresentou dificuldade em interpretar que a tonalidade mais opaca da imagem seria da neblina, e não do próprio elemento de superfície.

Nos modelos finais, observou-se que a mudança na configuração do parâmetro *learning rate*, permitindo que ele fosse atualizado ao longo das épocas, contribuiu para uma

melhor convergência do processo de otimização. Além disso, a aumentação de dados no conjunto de treino viabilizou que a performance no treinamento se tornasse mais coerente àquela da validação, o que conferiu maior generalidade ao modelo desenvolvimento. Já a variação nas dimensões das imagens permitiu que as perdas de informação dos *inputs* iniciais fossem minimizadas e que os pesos pré-treinados se tornassem mais compatíveis com o cenário avaliado desde o início do treinamento. Nesse contexto dos modelos finais, a ResNet50, que nos modelos iniciais havia proporcionado resultados inferiores à VGG16, passou a ser o modelo mais adequado para interpretar a maior complexidade dos dados, apresentando as melhores performances. Por fim, com construção dos modelos segregados e a variação do *threshold*, conseguiu-se obter uma modelagem que aumentou a capacidade de identificação dos elementos de superfície, ao mesmo tempo em que garantiu que, em cada predição, fosse classificado somente uma das condições atmosféricas.

Como consequência das melhorias, o modelo conseguiu aumentar a performance na maioria dos rótulos, reduzindo as ocorrências de inversões de classificação e passando a identificar casos que nos modelos iniciais não eram classificados, como nos elementos raros de superfície. Em contrapartida, mesmo no modelo final, em virtude do efeito da ausência de representatividade no conjunto de treinamento, as performances permaneceram baixas para os rótulos raros e para as classes *bare ground*, *cultivation* e *haze*. Nesse sentido, como próximos passos a serem explorados para melhorar a performance dessas classes, seria interessante aumentar a quantidade de imagens rotuladas. Sendo inviável essa opção, podem ser exploradas abordagens semi-supervisionadas para agregar à classificação conhecimento a partir das imagens não rotuladas. Seria interessante, também, explorar estratégias de *curriculum learning*, visando gerar um aprendizado dos rótulos de forma gradual, partindo das classes mais fáceis para as mais difíceis, que poderia contribuir para ganhos de performance adicionais nas classificações. Por fim, menciona-se que, além desses próximos passos que se associam a aspectos de modelagem, seria interessante, também, expandir a aplicação do modelo desenvolvido para além do conjunto de dados explorado, avaliando-o, por exemplo, em imagens extraídas do *Google Maps* ou de outras fontes, como o INPE ou a própria companhia *Planet*, associada às imagens utilizadas como base para o estudo.

Considerando os aspectos apresentados, entende-se que o trabalho atingiu o objetivo desejado de proporcionar uma classificação automática eficiência de rótulos de imagens de satélite da Amazônia, embora os próximos passos mencionados possam tornar os desenvolvimentos ainda mais robustos para aplicações reais de suporte à atuação de fotointerpretes especialistas no monitoramento do desmatamento.

REFERÊNCIAS

CHOLLET, F. **Deep learning with Python**. 2. ed. Nova York, Estados Unidos: Manning Publications Co., 2021.

DAVIES, E. R. **Computer vision: principles, algorithms, applications, learning**. 5. ed. Londres, Inglaterra: Academic Press, 2017.

FORSYTH, D.; PONCE, J. **Computer Vision: A Modern Approach.** 2. ed. Nova Jersey: Prentice Hall, 2011.

GÉRON, A. **Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow**. 3. ed. Sebastopol, Estados Unidos: O'Reilly Media, Inc., 2022.

GOLDENBERG, B. *et al.* **Planet: Understanding the Amazon from Space**. s.l: Kaggle, 2017. Disponível em: <https://kaggle.com/competitions/planet-understanding-the-amazon-from-space>.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep learning**. Londres, Inglaterra: The MIT Press, 2016.

INPE. **Metodologia utilizada nos sistemas PRODES e DETER - 2a Edição (Atualizada)**. São José dos Campos, SP: Coordenação de Ensino, Pesquisa e Extensão, 2022.

INPE. **Programa de monitoramento da Amazônia e demais biomas. Desmatamento – Amazônia Legal**. s.l: Coordenação Geral de Observação da Terra, 2023. Disponível em: <http://terrabrasilis.dpi.inpe.br/downloads/>.

SZELISKI, R. **Computer vision: algorithms and applications**. 2. ed. Suiça: Springer Nature, 2022.