UNIVERSIDADE FEDERAL FLUMINENSE

**André Fernandes Gonçalves**

# Classificação de Alagamento em Vias com Tráfego na Cidade do Rio de Janeiro através de Visão Computacional

NITERÓI

2025

**André Fernandes Gonçalves**

# Classificação de Alagamento em Vias com Tráfego na Cidade do Rio de Janeiro através de Visão Computacional

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Computação

Orientador:
Aura Conci

NITERÓI

2025

Ficha catalográfica automática - SDC/BEE
Gerada com informações fornecidas pelo autor

G635c Gonçalves, André Fernandes
Classificação de Alagamento em Vias com Tráfego na Cidade do Rio de Janeiro através de Visão Computacional / André Fernandes Gonçalves. - 2025.
78 f.: il.

Orientador: Aura Conci.
Dissertação (mestrado)-Universidade Federal Fluminense, Instituto de Computação, Niterói, 2025.

1. Visão Computacional. 2. Rede Neural Convolucional. 3. Classificação. 4. Alagamento. 5. Produção intelectual. I. Conci, Aura, orientadora. II. Universidade Federal Fluminense. Instituto de Computação. III. Título.

CDD - XXX

Bibliotecário responsável: Debora do Nascimento - CRB7/6368

ANDRÉ FERNANDES GONÇALVES

# CLASSIFICAÇÃO DE ALAGAMENTO EM VIAS COM TRÁFEGO NA CIDADE DO RIO DE JANEIRO ATRAVÉS DE VISÃO COMPUTACIONAL

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Computação

Aprovada em MARÇO de 2025.

BANCA EXAMINADORA

---

Prof. Aura Conci - Orientadora, UFF

---

Prof. Leandro Augusto Frata Fernandes, UFF

---

Prof. Flávia Cristina Bernardini, UFF

---

Prof. Aristófanes Corrêa Silva, UFMA

---

Prof. Raul Queiroz Feitosa, PUC

Niterói

2025

*Aos meus pais, que me apoiaram ao longo da minha jornada.*

# Agradecimentos

A minha orientadora, Aura Conci, que me ajudou em todo o caminho e sempre confiou em mim.

A UFF e a CAPES, pela bolsa que me ajudou durante meus estudos do mestrado.

A Luis Rezende e Otávio Flaeschen pela ajuda na criação do conjunto de dados.

Ao Centro de Operações do Rio pelo acesso as câmeras da cidade do Rio de Janeiro.

# Resumo

Este trabalho teve como objetivo desenvolver uma abordagem inicial para a classificação automática de alagamentos da cidade do Rio de Janeiro. Para isso, primeiramente foi criado um conjunto de dados original, o Rio Flooding Dataset (*RFD*), composto de 4620 imagens separadas em 78% para treino e 22% para validação. Este conjunto de dados foi montado com imagens de câmeras de ruas da cidade do Rio ao longo de diferentes meses, em diferentes horas do dia e com diversos níveis de alagamento. Após o conjunto de dados ser devidamente analisado e montado de forma equilibrada, cinco arquiteturas de redes neurais foram utilizadas nesse conjunto de dados. Essas arquiteturas foram escolhidas de acordo com trabalhos da literatura com temática relacionada ao problema abordado. Após o treinamento das arquiteturas *VGG*-19, *Inception*, *DenseNet*, *MobileNetV2*, e *Visual Transformer* para o problema de classificação de imagem, o *Visual Transformer* obteve melhor resultado. Este modelo, o *Visual Transformer*, foi analisado com diferentes níveis de iluminação e três diferentes otimizadores. Também foi investigado se o aumento na quantidade de imagens de treino traria ganhos compensadores. O desempenho dos resultados foi comparado com outros dois conjuntos de dados. Considerando todos os experimentos feitos, o *Visual Transformer* foi o melhor modelo com acurácia de 82,6% no conjunto de dados original. O código usado para treino e avaliação dos modelos estão disponíveis neste repositório de GitHub: https://github.com/afego/computervision. O conjunto de dados original está disponível em 10.5281/zenodo.15670835.

**Palavras-chave**: Visão Computacional, Rede Neural Convolucional, *Visual Transformer*, Classificação, Alagamento.

# Abstract

This work aimed to develop an initial approach for the automatic classification of floods in the city of Rio de Janeiro. To this end, this work first created an original dataset composed of 4620 images separated into 78% for training and 22% for validation. This dataset was assembled with images from street cameras in the city of Rio over different months, at different times of the day and at various flood levels. After the dataset was properly analyzed and assembled in a balanced way, five neural network architectures were evaluated on this dataset. These architectures were chosen according to works in the literature with a theme related to the problem addressed. After training the *VGG*-19, *Inception*, *DenseNet*, *MobileNetV2*, and *Visual Transformer* architectures for the image classification problem, the *Visual Transformer* obtained the best result. The *Visual Transformer* was then also analyzed with different lighting levels and three different optimizers. It was also investigated whether increasing the number of training images would bring compensating gains. The performance of the results was compared with two other datasets. Considering all experiments done, the *Visual Transformer* was the best model with an accuracy of 82.6% on the original dataset. The code used for training and evaluating the models is available in this GitHub repository: https://github.com/afego/computervision. The novel dataset from this research is available at 10.5281/zenodo.15670835.

**Keywords**: Computer Vision, Convolutional Neural Network, *Visual Transformer*, Classification, Flooding

# Lista de Figuras

# Lista de Tabelas

# Sumário

# 1 Introdução

A alta urbanização global, em conjunto com as rápidas mudanças climáticas globais, tem intensificado a ocorrência e o impacto de desastres ambientais. Dentre estes, alagamentos representam 40% dos desastres e, desde o ano 2000, desastres relacionados a alagamentos têm aumentado em 134% em comparação às décadas anteriores [84].

O Brasil, em específico, pode ter impacto anual de até 50 milhões de dólares em perdas econômicas sobre infraestrutura crítica vinda de alagamentos e ciclones [85]. Em 2024, o país registrou 140 ocorrências de desastres relacionados a alagamentos, desabrigando ou desalojando 30.000 brasileiros e trazendo prejuízos públicos e privados de 47 milhões de reais [13]. De acordo com análise da Defesa Civil do Rio de Janeiro, em 2019, mais de milhões de pessoas foram diretamente afetadas por inundações, alagamentos e enxurradas nas regiões da capital, metropolitana e baixada fluminense [14]. Em reportagem de O Globo, estima-se que as chuvas intensas no período de 2019 a 2023 causaram prejuízos de 2,9 bilhões de reais ao estado do Rio de Janeiro.

Segundo a Classificação e Codificação Brasileira de Desastres, alagamento é uma extrapolação da capacidade de escoamento de sistemas de drenagem urbana e consequente acúmulo de água em ruas, calçadas ou outras infraestruturas urbanas, em decorrência de precipitações intensas. Esse acúmulo afeta diretamente o trânsito de veículos e pedestres, principalmente em áreas urbanas, tornando imprescindível a vigilância de áreas possivelmente afetadas por alagamentos.

Parte da insuficiência do sistema de drenagem urbana do Rio de Janeiro vem do fato de que foi construído em grande parte no início do século XX, não sendo projetado para suportar os volumes de precipitação atualmente registrados, intensificados pelas mudanças climáticas e pela expansão urbana não planejada [67]. Essa insuficiência estrutural, aliada à falta de manutenção periódica, como desassoreamento de canais e limpeza de galerias pluviais, compromete a capacidade de escoamento das águas pluviais.

Diante deste panorama, fica claro que a mitigação dos impactos dos alagamentos no

estado do Rio de Janeiro exige não apenas intervenções estruturais, como melhorias no sistema de drenagem e planejamento urbano, mas também o desenvolvimento de ferramentas tecnológicas que permitam a detecção precoce, monitoramento eficiente e resposta rápida a esses eventos.

## 1.1 Contexto do problema

Para os órgãos governamentais responsáveis pelo monitoramento de áreas urbanas, como o Centro de Operações do Rio (COR), a observação constante dessas áreas urbanas é crucial para a emissão de alertas precoces e respostas eficazes. A detecção e avaliação oportuna dos riscos de inundação permitem que as autoridades implementem planos de evacuação, desloquem serviços de emergência e iniciem medidas de controle de inundação para minimizar danos e proteger vidas.

Atualmente, o monitoramento pelo COR é realizado na própria sala de controle, onde dezenas de funcionários monitoram uma quantidade ainda maior de telas que exibem as milhares de câmeras da cidade do Rio de Janeiro, como pode ser visto na Figura 1, tornando isto um trabalho árduo.

O monitoramento atualmente realizado pelo COR não é automatizado. Este monitoramento não só é dependente da mão-de-obra dos funcionários frente a diversas telas, como também depende em parte de notificações enviadas por terceiros, utilizando o aplicativo do COR para reportar problemas na cidade de forma direta.

## 1.2 Objetivo

Com base em diversas pesquisas sobre o problema das inundações urbanas utilizando diferentes Redes Neurais Convolucionais (CNN, do inglês *Convolutional Neural Network*), apresentadas na Seção 3.3, a presente dissertação visa desenvolver uma abordagem inicial utilizando *CNN*s com aprendizado por transferência (*transfer learning*), treinadas em um conjunto de dados original, para não apenas facilitar o monitoramento e classificação das milhares de câmeras na cidade do Rio de Janeiro pelo COR.

Portanto, esta pesquisa foi estruturada em quatro etapas principais: (i) aquisição e preparação dos dados, (ii) seleção das arquiteturas de redes neurais, (iii) treinamento e validação dos modelos, e (iv) avaliação do desempenho e análise dos resultados.


Figura 1: Sala central do Centro de Operações do Rio

Os dados foram obtidos por meio de câmeras do COR, enquanto a escolha das arquiteturas foi embasada em uma revisão criteriosa da literatura referente ao reconhecimento e classificação de alagamentos.

O treinamento e a validação foram conduzidos inicialmente por meio de validação simples, com o intuito de identificar o modelo de melhor desempenho. Posteriormente, esse modelo selecionado foi submetido a uma validação cruzada, proporcionando uma avaliação mais robusta e confiável dos seus resultados.

Finalmente, o desempenho do modelo foi comparado com resultados obtidos em outros conjuntos de dados, visando avaliar sua capacidade de generalização.

A seção seguinte apresenta uma descrição dos capítulos que abordam cada uma dessas etapas de forma aprofundada.

## 1.3 Organização do Texto

O capítulo 2 explica brevemente os conceitos mais importantes para o desenvolvimento deste trabalho, começando por técnicas de processamento de imagem e finalizando na abordagem dos fundamentos de redes neurais.

O capítulo 3 considera alguns trabalhos relevantes para o problema de classificação do estado de alagamento em regiões urbanas nos últimos 5 anos. Ao final, algumas características das tecnologias utilizadas em cada artigo, assim como as deste trabalho,

são comparadas em uma tabela.

O capítulo 4 descreve a metodologia usada na execução deste trabalho, explicando a criação do conjunto de dados, a escolha das arquiteturas de *CNN*s inicialmente empregadas e as métricas utilizadas para a comparação dos resultados.

O capítulo 5 apresenta e discute os resultados do treinamento das diferentes arquiteturas comparadas. O modelo com melhor performance é mais detalhadamente avaliado após a inclusão de alterações no pré-processamento e uso de diferentes conjuntos de dados de entrada.

O capítulo 6 sumariza as contribuições e limitações deste trabalho, apresentando conclusões relacionadas aos seus resultados. Também discute possíveis pontos para melhorias futuras em prosseguimentos da linha de pesquisa.

No Apêndice A estão citadas os códigos das câmeras do COR usadas e os números de imagens de cada classe delas armazenadas no conjuntos de dados inicial da pesquisa.

## 1.4 Contribuições e legados concretos desta dissertação

Até o momento com os resultados deste trabalho foram publicados no IEEE [32] contribuindo, também, com a disponibilização dos três conjuntos de dados (treino, validação e teste) correspondentes a organização que se fez nas diversas etapas desta dissertação e os programas de computador (fontes e executáveis) desenvolvidos. Esses se encontram publicamente disponíveis em 10.5281/zenodo.15670835.

# 2 Fundamentação Teórica

Este capítulo apresenta de forma sucinta os conceitos necessários para compreender as ferramentas utilizadas no capítulo 4. Para tal, o capítulo está dividido em duas seções: a primeira aborda conceitos fundamentais de processamento de imagem, enquanto a segunda seção é dedicada exclusivamente às redes neurais convolucionais e às arquiteturas empregadas.

## 2.1 Técnicas de Processamento de Imagens

Este trabalho foi desenvolvido em Python, utilizando a biblioteca *Pillow* para manipulação de imagens. As equações que definem as conversões entre os espaços de cor são apresentadas conforme descrito na documentação do *Pillow* [19].

### 2.1.1 Espaço de Cores

Uma imagem colorida é composta por uma ou mais bandas (ou canais). Usualmente, uma imagem digital colorida é formada por três bandas monocromáticas, sendo que cada banda armazena uma informação distinta da imagem colorida para o pixel p(m,n) naquela banda.

Cada banda, ou canal, de largura MM pixels e altura NN pixels, pode ser concebida como um plano bidimensional, onde um pixel é associado a um par de coordenadas inteiras p(m,n) pertencentes àquela banda. O valor armazenado na posição (m,n) indica a intensidade da banda naquele ponto da imagem, geralmente codificada em 256 níveis possíveis (valores de 0 a 255) armazenados em um byte. Nesse contexto, o valor 0 corresponde à intensidade mínima (preto ou ausência de luz) e 255 à intensidade máxima (branco ou máxima luminosidade) [52, 76].

Existem diferentes espaços para representação de cores, sendo o mais comum o RGB-CIE, definido pela *Commission Internationale de l'Éclairage* (CIE) em 1931, ou simples-

mente RGB (do inglês *Red, Green and Blue*). Neste espaço, cada cor é representada em seu componente espectral primário correspondente, de modo que cada banda armazena exclusivamente a intensidade do vermelho, verde ou azul, respectivamente [33, 52].

Outro espaço importante é o XYZ-CIE 1931 (abreviado como XYZ), projetado para representar todas as cores visíveis, incluindo aquelas que não podem ser representadas no espaço RGB. As componentes XX e ZZ correspondem a elementos sem correspondência direta na percepção visual humana, não existindo, portanto, como cores perceptíveis na prática. Já a componente YY representa a luminância, ou seja, a melhor representação monocromática possível da cena originalmente colorida [60].

### 2.1.2 Conversões entre RGB e YCbCr

Nas conversões entre espaços de cor, não ocorre qualquer alteração prévia nos valores dos pixels no espaço de cor original antes de serem convertidos para o espaço de cor desejado.

As equações 2.1, 2.2 e 2.3 apresentam a transformação do espaço RGB para o espaço YCbCr. Por sua vez, as equações 2.4, 2.5 e 2.6 descrevem a conversão inversa, do espaço YCbCr para RGB.

$$Y \leftarrow R \cdot 0{,}29900 + G \cdot 0{,}58700 + B \cdot 0{,}11400 \tag{2.1}$$

$$Cb \leftarrow R \cdot -0{,}16874 + G \cdot -0{,}33126 + B \cdot 0{,}50000 + 128 \tag{2.2}$$

$$Cr \leftarrow R \cdot 0{,}50000 + G \cdot -0{,}41869 + B \cdot -0{,}08131 + 128 \tag{2.3}$$

$$R \leftarrow Y + (Cr - 128) \cdot 1.40200 \tag{2.4}$$

$$G \leftarrow Y + (Cb - 128) \cdot -0{,}34414 + (Cr - 128) \cdot -0{,}71414 \tag{2.5}$$

$$B \leftarrow Y + (Cb - 128) \cdot 1{,}77200 \tag{2.6}$$

A conversão para YCbCr foi realizada com o intuito de utilizar o canal Y para corrigir a intensidade luminosa das imagens, como comentado na seção 4.1.3.

### 2.1.3 Transformações de Intensidade no Domínio Espacial

As transformações de intensidade são funções matemáticas no domínio espacial da imagem, que operam diretamente no valor dos pixels da imagem de entrada [33]. Essas

funções podem ser representadas pela expressão:

$$g(x,y) = T\,[f(x,y)] \tag{2.7}$$

onde *f(x,y)* é a imagem de entrada, *g(x,y)* é a imagem de saída e $T$ é uma função definida sobre um ponto *(x,y)* ou sua vizinhança.

Uma das transformações de intensidade mais simples é a transformação linear, onde os valores de intensidade de um canal são multiplicados por uma constante, como representado pela equação:

$$i^{'} = \alpha \times i \tag{2.8}$$

Através da equação 2.8, a intensidade $i$ de um pixel é multiplicada pelo fator de escala $\alpha$, resultando na nova intensidade $i^{'}$.

### 2.1.4 Filtragem por Convolução

Também denominados máscaras ou kernels, os filtros consistem em matrizes de dimensão $m \times n$ que são aplicadas sobre uma imagem por meio de operações de multiplicação elemento a elemento entre os valores da máscara e os pixels correspondentes da imagem, seguidas de somas dos resultados. O valor obtido a partir dessa operação corresponde ao pixel processado na imagem resultante.

As dimensões e os valores que compõem essas máscaras variam de acordo com a característica específica que se deseja extrair de cada vizinhança $m \times n$ da imagem, tais como bordas, texturas ou outras propriedades locais [76].

## 2.2 Redes Neurais Convolucionais

As redes neurais convolucionais, ou *CNN*s, são uma classe especial de redes neurais artificiais que se especializam em processar dados com uma topologia em matrizes [34].

Inspiradas pela organização do córtex visual dos mamíferos [41], as *CNN*s são projetadas para extrair características locais que dependem de uma pequena vizinhança na imagem, onde estas pequenas características podem ser usadas por outras camadas para detectar características de maior ordem e extrair informações sobre a imagem [7].

*CNN*s são usadas principalmente para tarefas que trabalham em imagens, como reconhecimento de objetos, segmentação de imagens e detecção de padrões complexos de

dados visuais.

O aprendizado de características locais permite que a *CNN* reconheça esses padrões em qualquer outro local da imagem, tornando-a invariante a translação. A utilização de características de uma camada por camadas sucessivas permite que a *CNN* aprenda padrões espaciais hierárquicos [18].

### 2.2.1 Arquitetura de uma Rede Neural Convolucional

A arquitetura básica de uma *CNN* inclui três tipos principais de camadas: convolucional, de pooling e totalmente conectada. Uma *CNN* típica é formada por pares de camadas de convolução e de pooling empilhadas, que ao final são ligadas a uma camada totalmente conectada. A Figura 2 mostra uma arquitetura genérica de *CNN*.

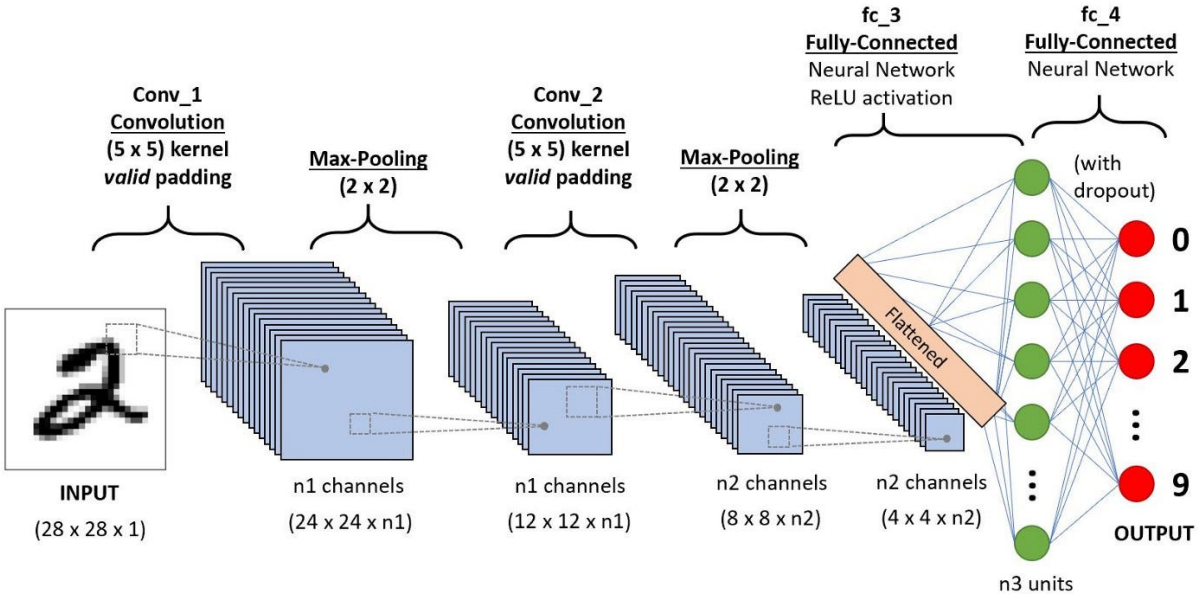


Figura 2: Arquitetura de uma *CNN* genérica.

#### 2.2.1.1 Camada Convolucional

A camada convolucional é o núcleo de uma *CNN* e é responsável pela extração de características. Para isso pesos na forma de kernels, ou filtros da Análise de Imagem tradicional, são aplicados sobre a imagem, onde são escorregados e multiplicados pelos valores dos pixels da imagem de entrada e somados resultando em um pixel para cada kernel, usando uma quantidade definida de passos.

Esse processo gera os mapas de características, onde cada mapa representa a resposta de um filtro específico a uma região da imagem de entrada. Essa camada detecta padrões locais como bordas, texturas ou formas, (de acordo com o kernel usado) que podem ser usados para extrair informações de padrões mais complexos conforme a rede se aprofunda.

A operação de convolução S entre um filtro I e uma entrada K pode ser representada como:

$$S(x,y) = \sum_{m} \sum_{n} I(x+m,y+n)K(m,n) \tag{2.9}$$

onde S(x,y) é o mapa de características gerado (saida), I(x,y) representa a entrada local (imagem original) e K(m,n) o kernel aplicado.

#### 2.2.1.2 Camada de Pooling

A camada de pooling altera a saída da camada convolucional com um resumo estatístico de uma vizinhança, melhorando a eficiência computacional da rede pela diminuição da dimensionalidade da saída e ajudando a tornar a saída menos variante a translações da entrada. Este resumo estatístico pode ser feito com diferentes operações. Por exemplo a arquitetura VGG19 [78] utiliza *max pooling*, onde a saída é o maior valor na vizinhança analisada. A operação *average pooling* é utilizada no InceptionV3 [81], onde a média da vizinhança é calculada.

#### 2.2.1.3 Camada Totalmente Conectada

Estas são geralmente as últimas camadas em uma *CNN*, sendo responsáveis pela combinação das características extraídas nas camadas anteriores para realizar a classificação final. Nesta camada, todos os neurônios estão conectados a cada unidade da camada anterior, como um perceptron multicamadas comum [58]. O seu aprendizado é feito por um algoritmo de retro-propagação de erro [70].

### 2.2.2 Processo de Treinamento de uma CNN

O treinamento de uma *CNN* é realizado por meio de aprendizado supervisionado, onde o modelo ajusta os pesos e vieses das camadas convolucionais (totalmente conectadas) para minimizar a diferença entre as previsões e os rótulos verdadeiros.

A função de perda, que calcula o erro da rede, é derivada e propagada para ajustar

os pesos de forma iterativa. Este trabalho utilizou a Entropia Cruzada como função de perda, que é definida por:

$$H(P,Q) = -E_P[logQ] \tag{2.10}$$

onde H(P,Q) é a entropia cruzada da distribuição Q em relação a distribuição P, sendo a distribuição Q o resultado da *CNN* e a distribuição P a distribuição de treinamento [54]. $E_P[.]$ é o valor esperado em relação a distribuição P.

Para minimizar a função de perda, foi usada a descida de gradiente estocástica (*SGD*, do inglês *Stochastic Gradient Descent*) [69]. Enquanto a descida de gradiente padrão calcula o gradiente utilizando todo o conjunto de dados, o *SGD* utiliza amostras de dados para o cálculo do gradiente na iteração atual. Dessa forma, a *SGD* altera o valor dos parâmetros da seguinte forma:

$$\omega^{\tau+1} = \omega\tau - \eta\Delta E \tag{2.11}$$

onde $\omega$ é o peso sendo usado, $\tau$ é o número da iteração, $\eta$ é a taxa de aprendizado e $\Delta E$ é o gradiente do erro.

### 2.2.3 Aprendizado por Transferência

Uma possibilidade importante nas *CNN*s é o uso de redes pré-treinadas e aprendizado por transferência (*transfer learning*), onde os pesos dos modelos pré-treinados em grandes conjuntos de dados são aproveitados e ajustados para problemas específicos com menos dados, reduzindo o tempo de treinamento e melhorando a precisão em aplicações com conjuntos de dados menores. Neste trabalho, o aprendizado por transferência baseado no IMAGENET [24] foi aplicado em todos os modelos avaliados, alterando sua última camada para uma classificação binária.

### 2.2.4 Visual Geometry Group (VGG)

A *VGG* é uma família de *CNN*s que possuem arquiteturas simples. Propostas por [78], as *VGG*s são compostas por blocos uniformes empilhados com duas camadas convolucionais e uma camada de *pooling*. As camadas de *pooling* têm tamanho $2 \times 2$ reduzindo a dimensionalidade da convolução pela metade.

Já as camadas convolucionais compartilham do mesmo tamanho de filtro, sendo estes $3 \times 3$. Além disso, cada bloco dobra a quantidade de filtros de suas camadas convolucionais

em relação ao bloco anterior. Por exemplo, o primeiro bloco de uma rede *VGG* possui 64 filtros $3 \times 3$ em suas camada convolucionais, enquanto o segundo bloco possui 128 filtros $3 \times 3$.

Este trabalho utilizou a *VGG*-19, que indica a existência de 19 camadas de aprendizado, contando não só as camadas convolucionais como as camadas completamente conectadas.

### 2.2.5 Inception

Diferente do *VGG*, a família *Inception*[80] de *CNN*s utiliza o empilhamento de módulos Inception, onde convoluções de diferentes tamanhos são realizadas no mesmo módulo, especificamente convoluções de $1 \times 1$, $3 \times 3$ e $5 \times 5$. Ao final, tais convoluções são concatenadas junto ao *pooling* da própria entrada, gerando a saída do módulo.

O *InceptionV3*[81] fatoriza convoluções maiores em uma sequência de convoluções menores. A convolução $5 \times 5$, por exemplo, é substituída por duas convoluções $3 \times 3$, diminuindo a quantidade de parâmetros mas mantendo a expressividade das características extraídas.

Outra modificação implementada no *InceptionV3* foi a utilização do *label smoothing*, onde a confiança do modelo é reduzida para evitar *overfitting*. Isso é feito alterando a probabilidade esperada de cada classe:

$$q(k) = (1 - \epsilon)\delta_{k,j} + \frac{\epsilon}{K} \tag{2.12}$$

Onde $j$ é a classe correta, $k$ é a classe sendo avaliada, $\delta_{k,j}$ é o *Delta de Dirac*, que se torna 1 quando $k = j$ e 0 em qualquer outro caso, $\epsilon$ é uma probabilidade de incerteza do modelo, e $K$ é o número de classes. Portanto, o *label smoothing* retira da probabilidade *1* da classe correta uma incerteza $\epsilon$ e a distribui igualmente entre as outras $K$ classes do problema.

### 2.2.6 DenseNet

A característica de maior importância da *DenseNet*[40] é que sua arquitetura é baseada em blocos densos, que são compostos por camadas convolucionais onde sua entrada recebe a saída de todas as camadas convolucionais anteriores a ela. Os blocos densos são ligados por camadas de transição, compostas por uma camada de convolução $1 \times 1$ e uma camada

de *average pooling* 2×2. A arquitetura de uma *DenseNet* simples pode ser vista na Figura 3 [40].

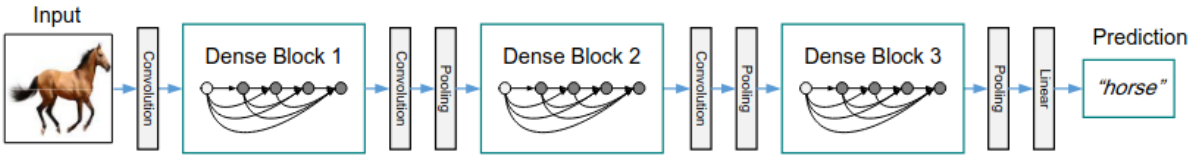


Figura 3: Arquitetura de uma *DenseNet* genérica[40].

### 2.2.7 MobileNet

A família *MobileNet* de redes neurais foi desenvolvida para utilização por aplicações móveis ou embarcadas [38], posuindo arquiteturas mais simples e leves em comparação a outras redes neurais profundas, mas continuam a oferecer boa performance. Sua maior distinção de outras *CNN*s é a introdução de convoluções separáveis em profundidade, que substituem as convoluções tradicionais.

As convoluções separáveis em profundidades fatorizam uma convolução comum em uma convolução em profundidade e uma convolução 1×1, também chamada de convolução pontual. Isso significa que na convolução em profundidade, em vez de aplicar um filtro que abrange todos os canais da janela a qual está realizando a convolução, será aplicada um filtro por canal, onde a saída de cada canal é combinada pela convolução pontual, reduzindo o custo computacional envolvido na convolução.

Além disso, a arquitetura também possui dois hiperparâmetros globais, um multiplicador de largura $\alpha$ e um multiplicador de resolução $\rho$, que servem para reduzir o custo computacional da rede. O hiperparâmetro $\alpha$ serve para afinar a rede neural em cada camada por um fator $\alpha$, aplicado tanto no número de canais de entrada quanto de canais de saída. Já o hiperparâmetreo $\rho$ é aplicado na imagem de entrada e nas representações internas de cada camada, multiplicando-os por $\rho$.

A arquitetura *MobileNetV2*[71] introduz blocos inversos residuais na arquitetura da rede. Este bloco aumenta a dimensionalidade da entrada através de um conjunto de convoluções 1×1 e aplica uma convolução em profundidade em seguida, para capturar informações em cada canal expandido. Ao final, uma convolução 1×1 é aplicada para reduzir os canal de volta ao tamanho original.

### 2.2.8 Visual Transformer (ViT)

A arquitetura *Visual Transformer*[48] aplica o conceito de *transformers*[87] utilizados em processamento de linguagem natural para tarefas de classificação de imagens, como alternativa às *CNN*s. A arquitetura em si é relativamente simples, composta um *transformer enconder*, sem a utilização de um *transformer decoder*, que fornece sua saída a um perceptron multicamadas simples (*MLP*, do inglês *Multilayer Perceptron*), sendo este o responsável pela classificação. Apesar de simples, nos processos necessários para enviar a imagem para o *transformer* é onde a complexidade da arquitetura aparece.

A Figura 4 [48], mostra a arquitetura de um *ViT*. Diferente de uma *CNN* comum, a imagem é dividida em blocos de tamanho fixo denominados *patches*. Esses *patches* são transformados em vetores e linearizados. Para convertê-los ao espaço de características do modelo, os *patches* passam por um *embedding*, através de uma projeção em um espaço de dimensão fixa. Antes de enviar a sequência de *embeddings* ao *transformer*, é necessário adicionar um *token* de posicionamento para informar à rede sobre a posição de cada *patch* na imagem original. Além desses *token*s, um *token* adicional *class* é colocado no início da sequência, que receberá a saída do processamento realizado pelo *transformer*.

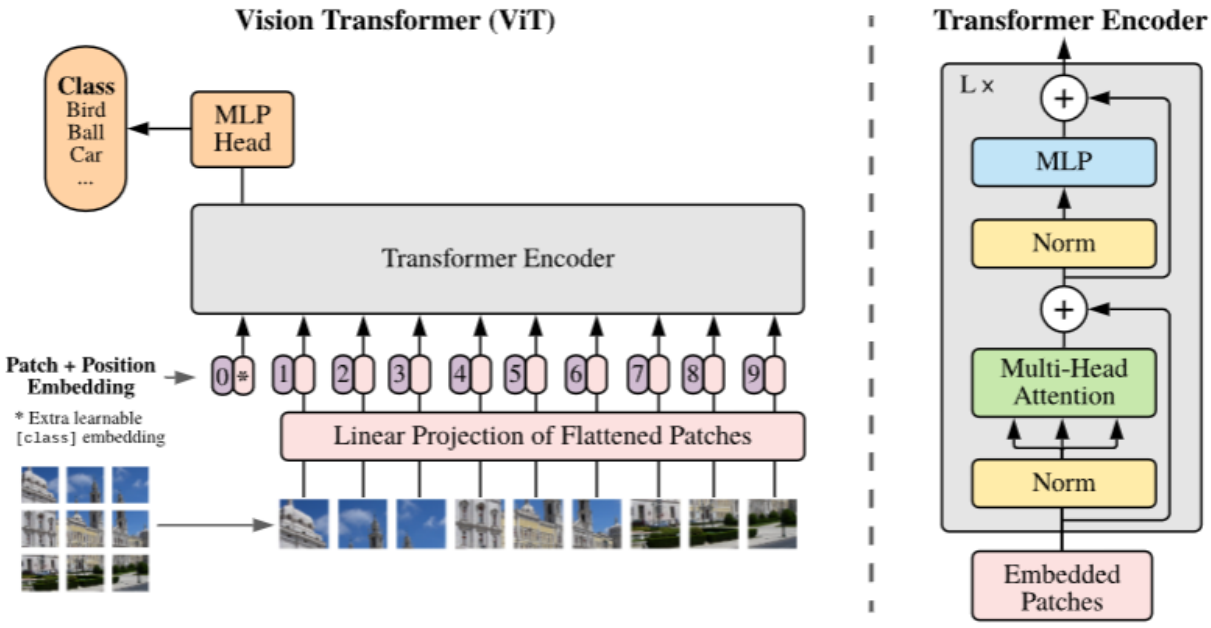


Figura 4: Arquitetura de um *Visual Transformer*[48].

O *Transformer* é uma sequência de pares de camadas alternadas, o tamanho dessa sequência de camadas define a profundidade do modelo. O primeiro par de camadas consiste em uma camada de normalização, transformando os valores de todos os elementos da sequência para limites predefinidos, e uma camada de *Multi-Headed Self-Attention*,

onde é computada uma soma ponderada de entre cada par de elemento do *patch*. O segundo par de camadas possui uma camada de normalização e uma camada de *MLP*, para processar cada elemento do *patch*.

O processamento final do *transformer* é enviado para um *MLP* final denominado *MLP Head*, responsável por receber o *token class* que representa as informações da imagem aprendidas no *transformer*, e utilizá-lo para gerar a classificação final da imagem.

# 3 Trabalhos Relacionados

Diversos estudos vêm sendo realizados na área de classificação e detecção de inundações, utilizando tanto algoritmos tradicionais de aprendizado de máquina quanto redes neurais profundas. Tais pesquisas empregam diferentes fontes de dados, incluindo imagens capturadas por satélites, imagens provenientes de câmeras fixas e, ainda, dados numéricos obtidos a partir de instrumentos de medição, como pluviômetros, entre outros.

Neste capítulo, inicialmente serão apresentados trabalhos que empregaram algoritmos de aprendizado de máquina para prever eventos de alagamento. Em seguida, serão brevemente descritos estudos que utilizaram imagens de satélite para a análise de áreas alagadas, incluindo uma seção dedicada à aplicação de arquiteturas do tipo *U-Net* para segmentação e classificação dessas imagens.

Por fim, a última seção será dedicada a pesquisas cujo foco principal é a classificação de imagens relacionadas a alagamentos urbanos. Devido ao foco desta dissertação estar centrado na classificação de imagens capturadas por sistemas de câmeras urbanas, essa seção será discutida com maior profundidade em relação às anteriores.

Para facilitar a compreensão da evolução do campo, os trabalhos apresentados em cada seção serão organizados em ordem cronológica, iniciando pelos mais antigos.

## 3.1 Aprendizado de Máquina

Utilizando dados disponibilizados pelo Departamento Meteorológico de Bangladesh (*Bangladesh Meteorological Department*) e pelo Conselho de Desenvolvimento Hídrico de Bangladesh (*Bangladesh Water Development Board*), Toufique *et al.* [83] treinaram quatro algoritmos de aprendizado de máquina para a predição de inundações na região. Entre os modelos avaliados, o *Random Forest* [10] apresentou o melhor desempenho, alcançando uma acurácia de 85,6%.

Jamunadevi *et al.* [44] propuseram uma abordagem híbrida que combinou uma rede

neural recorrente do tipo memória de curto e longo prazo (LSTM) [37] com uma rede neural convolucional (*CNN*), visando a previsão de alagamentos ao longo de um período de um ano no estado de Tamil Nadu, Índia. Os dados utilizados para treinamento englobaram variáveis meteorológicas e ambientais, tais como temperatura, umidade, precipitação, velocidade do vento, radiação solar e evapotranspiração. O modelo desenvolvido obteve uma acurácia de 97,4% na previsão dos eventos de alagamento ocorridos em 2024.

Vimala *et al.* [88] empregaram um conjunto de 22 indicadores ambientais para realizar a predição de alagamentos, utilizando seis algoritmos de aprendizado supervisionado, incluindo o *Random Forest*. Entre os modelos testados, a Regressão Logística [23] foi o que apresentou o melhor desempenho, com uma acurácia de 99,85%, enquanto o *Random Forest* atingiu uma acurácia de 89,37%.

Shah *et al.* [75] utilizaram os algoritmos *Random Forest* e *XGBoost* [17] para gerar um mapa de suscetibilidade a inundações da cidade de Chennai, também no estado de Tamil Nadu. Os autores empregaram seis métricas provenientes de três fontes distintas para treinar os modelos, nos quais o *XGBoost* apresentou a melhor acurácia, alcançando 70%.

## 3.2 Imagens de Satélite

Kool *et al.* [50] classificaram o estado de inundação sazonal de um pântano em Mara na Tanzânia. Para isso, treinaram um *Random Forest* sazonalmente em conjuntos de imagens do satélite Sentinel-2 [21], obtendo acurácia de 98,6%.

Siddique *et al.* [77] propuseram uma abordagem integrada utilizando *Random Forest*, *k-Nearest Neighbours* (*k-NN*)[22] e *k-means*[42] para classificar imagens do satélite Sentinel-1 [20] do estado da Índia de Uttar Pradesh. Obtendo acurácia de 97,0% para a cidade de Basti e 93,8% para a cidade de Ayodhya, ambas no estado de Uttar Pradesh.

Reddy e Vimal [65] classificaram 400 imagens de satélites, obtendo resultados de 92,72% de acurácia com análise discriminante linear [82] e 95,05% com AlexNet [51].

Cao *et al.* [12] utilizaram uma *CNN* simples, proposta pelos autores, onde a saída da camada completamente conectada alimenta um classificador de Máquina de Vetores de Suporte (*SVM*, do inglês *Support Vector Machine*) [29] para classificar o pântano de Qilihai, na China. Obtiveram acurácia de 90,3% no conjunto de dados, que foi montado através de imagens do Sentinel-1 e Sentinel-2.

Myin e Thein [59] aplicaram um *Random Forest* em 1000 imagens do Sentinel-1 da região de Bago, em Myanmar, com acurácia de 93,18%.

Huang *et al.* [39] projetaram um modelo chamado *WaterDetectionNet* (WDNet) para mapear alagamentos, usando imagens de radar de abertura sintética [25] do Sentinel-1 sobre lago de Poyang, na China. O modelo WDNet com módulo de atenção própria atingiu acurácia de 98,7%.

Bouchelkia *et al.* [8] propuseram um abordagem para dectectar diferenças em terrenos pós-inundações, com pares de imagens do Sentinel-2 antes e depois da ocorrência de inundações na cidade de Derna na Líbia. A arquitetura de rede neural profunda [73] proposta pelos autores obteve área sobre união de 95,8%.

Sudiana *et al.* [79] classificaram imagens do Sentinel-1 de áreas urbanas em Jacarta, capital da Indonésia, utilizando uma *3D-CNN* [64]. Conseguiram acurácia de 71,8% com 20 imagens de satélite.

### 3.2.1 U-Net

Com o objetivo de segmentar e classificar imagens de alagamento de diferentes fontes, Ahmed *et al.* [2] compararam diferentes arquiteturas para esses trabalhos: *DeepLabv3*[16]; *MaskRCNN* [35]; e *U-Net* [68]. O DeepLabv3 obteve interseção sobre união [66] médio de 0,95, em comparação com 0,93 do *MaskRCNN* e 0,94 do *U-Net*.

Alhady *et al.* [4] compararam a performance do *U-Net* em imagens do Sentinel-2 com diferentes métodos de índice de água [53], onde a utilização do *Modified Normalized Difference Water Index* [90] obteve a melhor performance, com área sobre união de 97,71%/.

Chandram *et al.* [15] propuseram um sistema de detecção de alagamentos baseado no U-Net com área sobre união de 0,85.

Pech-May *et al.* [62] classificaram imagens do Sentinel-1 do estado mexicano de Tabasco, utilizando o *U-Net*. Os autores compararam a acurácia de área sobre união treinando em diferentes épocas, onde o modelo treinado em 200 épocas foi o melhor, com acurácia de 94,31% e área sobre união de 0,730.

## 3.3 Alagamento em áreas urbanas

Akshya e Priyadarsini [3] utilizaram imagens aéreas de áreas alagadas na Índia para classificar o grau de severidade de alagamentos. Inicialmente, é realizada a extração de características utilizando *Bag of Visual Words* ($BoVW$). Depois uma clusterização com *k-means*. Na classificação eles usaram uma abordagem híbrida, feita através de uma $SVM$. Usando um conjunto de dados de 200 imagens, sendo 100 para cada classe, a abordagem conseguiu acurácia média de 92%. Apesar da abordagem interessante de utilizar $BoVW$ para classificação com aprendizado de máquina clássico, os autores somente avaliaram $SVM$ com diferentes *kernels*, sem comparar com outros algoritmos de aprendizado de máquina. Além disso, o modelo não só foi testado em somente um conjunto de dados, como o conjunto de dados usado não foi disponibilizado.

Sazara *et al.* [72] abordaram o problema de classificação de inundação (ou não em imagens) utilizando a saída de um extrator de características para treinar modelos de aprendizado de máquina diferentes. Para extração de características foram avaliados Padrões Locais Binários (LBP) , Histograma de Gradientes Orientados (HOG) e uma rede neural $VGG$-16. Os modelos de aprendizado de máquina treinados nas características extraídas foram a Regressão Logística, k-NN, e uma Árvore de Decisão. A combinação que forneceu melhores resultados, com precisão de 0,94 e revocação de 0,97 , foi o uso da $VGG$-16 como extrator de características e a Regressão Logística como classificador. O conjunto de dados utilizado consistiu de 253 imagens de alagamento e 238 sem alagamento foi disponibilizado e foi usado para comparação do melhor modelo desta pesquisa, na Seção 5.5. Entretanto, os autores não avaliram o modelo em outros conjuntos de dados.

Jony *et al.* [45] classificaram binariamente alagamentos em imagens retiradas de redes sociais. Para isso, utilizaram três diferentes $CNN$s para extração de características com uma simples rede neural para classificação binária do problema, usando duas abordagens, sobre o conjunto de dados do *Disaster Image Retrieval from Social Media* 2017[56]. A primeira abordagem utilizou InceptionV3 e Xception como extratores, enquanto a segunda abordagem utilizou $VGG$-16 como extrator. Os resultados mostraram que realizar uma média dos resultados de cada classe entre as duas abordagens gerou melhor acurácia que usar cada abordagem unicamente. As abordagens em conjunto, a primeira abordagem e segunda abordagem tiveram acurácia de 93% , 92,8% e 90,6% respectivamente. Este trabalho utilizou o conjunto de dados do desafio *MediaEval 2017* e comparou seus resultados com outros métodos apresentados no mesmo desafio, obtendo resultados competitivos com os melhores deste desafio.

Vineeth e Neeba [89] desenvolveram um método para calcular a profundidade de alagamentos e classificar se há alagamento ou não em um conjunto de dados de aproximadamente 5000 imagens. Para classificação, foi utilizada uma MobileNet, alcançando acurácia de 0,94. A utilização de *VGG*-16 para o cálculo de profundidade, separada em 4 classes, resultou em acurácia de 0,78. Este trabalho não disponibilizou o conjunto de dados e também não comparou seus resultados com outros modelos ou em outros conjuntos de dados.

Piedad *et al.*[63] propuseram um sistema de detecção de alagamento utilizando imagens de **circuito fechado** de câmeras. O conjunto de dados utilizado consiste de 30000 imagens separadas em dia e noite. Primeiramente, as imagens passam por análise de cena utilizando o DeepLabv3, um modelo de aprendizado profundo e segmentação semântica, onde objetos são detectados e são aplicadas cores diferentes em tais objetos para facilitar contagem e visualização. As imagens preprocessadas com essas cores foram utilizadas para o treino e teste de uma *CNN*, que obteve 80,67% de acurácia de 81% de revocação. Em termos de acurácia entre dia e noite, enquanto a acurácia para imagens de dia ficou com média de 87,08%, a acurácia para imagens de noite ficou com média de 70,66%. O modelo não foi testado em outros conjuntos de dados ou comparados com outras arquiteturas. O conjunto de dados também são foi disponibilizado.

Pally e Samadi[61] desenvolveram um pacote em *Python* chamado *Flood Image Classifier* para classificar e detectar objetos em imagens de inundações. O pacote consiste em diversas arquiteturas de *CNN*s (Mask RCNN, Fast RCNN, SSD MobileNet, EfficientDet, YOLOv3) treinadas em mais de 9000 imagens. Também utilizam detecção de borda através do algoritmo de Canny[11] para estimar o nível de água e **técnicas de segmentação** para identificar e medir a inundação, sendo possível avaliar a profundidade e gravidade dos danos. Os autores disponibilizaram os modelos de detecção no repositório de GitHub.

Com o objetivo de detectar se ocorreu um desastre, e se há fogo ou inundação, Sghaier *et al.*[74] utilizaram uma simples *CNN* de três camadas de convolução para classificar a imagem em 4 classes: se há fogo, se não há fogo, se há inundação; e se não há inundação. Com um conjunto de dados de 1045 imagens contendo fogo, 231 imagens sem fogo, 1034 imagens com inundação e 326 imagens sem inundação, a *CNN* obteve acurácia de 0,99. Vale apontar que ambas as categorias de não haver fogo e de não haver inundação representam situações de normalidade, portanto, são redundantes e poderiam ser uma única classe, resumindo o estudo para um problema de três classes. Também não houve nenhum tipo de comparação com outras arquiteturas ou conjuntos de dados, além do

conjunto utilizado não ser disponibilizado.

Em um conjunto de dados de 2000 imagens, Agung *et al.*[1] utilizaram uma MobileNet com três novas camadas antes da camada completamente conectada para realizar a classificação binária de alagamento em ruas. Os autores obtiveram acurácia de 0,96, precisão de 0,95 e revocação de 0,97. Entretanto, estes resultados não foram comparados com nenhuma outra arquitetura, inclusive com a própria MobileNet sem as alterações realizadas. Assim como também não houve comparação com outros conjuntos de dados.

Islam *et al.*[43] propuseram uma abordagem combinando *CNN* e algoritmos de ordenação para classificar imagens de drones em três diferentes graus de severidade. O sistema criado detecta o nível de alagamento e calcula a distância do drone até a área afetada, gerando então um valor ponderado representando a prioridade de atenção do drone à cada área detectada, estes valores são ordenados pelo algoritmo de ordenação. Com 1011 imagens divididas entre treino, teste e validação, os modelos testados *DenseNet* e *InceptionV3* conseguiram acurácia de 81% e 83%, respectivamente. Como reconhecido pelos próprios autores, mais estudos sobre a viabilidade do método descrito precisam ser feitos, visto que não houve comparação com outros conjuntos de dados.

Hidayat *et al.*[36] compararam a performance do *VGG*-16 e *MobileNet* para classificar o alagamento em rodovias. Com um conjunto de 2000 imagens da cidade de Macáçar, na Indonésia, os autores obtiveram acurácia de 99% e tempo de processamento de 9 segundos com o *MobileNet*. Já o *VGG*-16 conseguiu acurácia de 96% e tempo de processamento de incríveis 69 segundos. Os autores não especificaram a configuração da máquina onde os modelos foram treinados, mas justificaram a diferença discrepante entre o processamento dos modelos através da simplicidade do *MobileNet* em comparação ao *VGG*-16. Não houveram avaliações em outros conjuntos de dados, o conjunto de dados utilizado não foi disponibilizado ou descrita a sua quantidade de imagens.

Arora e Bhavadharini [5] compararam as arquiteturas *VGG*-19 e *VGG*-16 para classificar cenas de desastres naturais, mais especificamente ciclones, terremotos, inundações e incêndios florestais. Treinando os modelos em 48 épocas, o modelo *VGG*-19 conseguiu acurácia de 94,0% e o *VGG*-16 conseguiu 92,1%, com o total de aproximadamente 4000 imagens. Apesar da alta acurácia, os autores não especificaram a acurácia por classe. Essa é uma análise importante pois o conjunto de treino possui 3321 imagens, onde a classe de terremoto possui 1012 imagens, enquanto a classe de ciclone possui somente 696 imagens, mostrando um claro desequilíbrio. Outras arqutieturas não foram avaliadas, assim não houve testes em outros conjuntos de dados ou a disponibilização do conjunto de dados

Tabela 1: Resumo dos trabalhos telacionados

| Trabalho | Extração de Característica | Classificação | Quantidade de Imagens |
|---|---|---|---|
| Esta pesquisa | Não | VGG-19, InceptionV3, DenseNet, MobileNetV2, ViT | 4620 |
| Akshya | BoVW | k-means e SVM | 200 |
| Sazara et al. | VGG16 | Regressão Logística | 491 |
| Jony et al. | InceptionV3, Xception, VGG16 | CNN própria | Não Informado |
| Vineeth e Neeba | Não | MobileNet | 5000 |
| Piedad et al. | Não | CNN própria | 30000 |
| Pally e Samadi | Não | Mask RCNN, Fast RCNN, SSD MobileNet, EfficientDet, YOLOv3 | 9000 |
| Sghaier et al. | Não | CNN própria | 1045 |
| Agung et al. | Não | MobileNet | 2000 |
| Islam et al. | Não | DenseNet e InceptionV3 | 1011 |
| Hidayat et al. | Não | VGG16 e MobileNet | Não Informado |
| Arora e Bhavadharini | Não | VGG16 e VGG19 | 4000 |

utilizado.

A Tabela 1 mostra um resumo da literatura sobre classificação de estados de inundações apresentada neste capítulo.

# 4 Materiais e Métodos

Este capítulo descreverá todo o processo para a criação da metodologia usada. Desde a descrição e adaptação do conjunto de dados utilizado, cobrindo as escolhas dos modelos de classificação até a forma de avaliação dos resultados, além das tecnologias utilizadas em cada passo.

Apesar de ser possível classificar a situação de alagamento de uma rua em diferentes níveis, para o problema de classificação descrito, as diferentes situações foram simplificadas para um problema binário, onde o objetivo é definir se a quantidade de água na rua permite o trânsito de veículos normalmente (classe 0), ou se a quantidade é suficiente para impactar o trânsito (classe 1).

## 4.1 Conjunto de dados

O conjunto de dados consiste em imagens de 77 câmeras diferentes da cidade do Rio de Janeiro, capturadas pelo sistema de câmeras do Centro de Operações do Rio (COR). Neste conjunto, cada câmera possui 30 imagens para cada uma das classes deste problema de classificação binário. Ou seja, o conjunto de dados é formado por 60 imagens de cada câmera, totalizando 4.620 imagens. Do total de 77 câmeras, 60 foram escolhidas para compor o conjunto de treino e 17 câmeras para validação, ou seja, 78% das câmeras compõem o conjunto de treino e 22% o conjunto de validação. Esses 3 conjuntos estão disponíveis em 10.5281/zenodo.15670835.

Este trabalho frequentemente se refere ao conjunto de dados usado pelo número de câmeras, em vez de descrever diretamente o número de imagens. Isso é feito propositalmente, visto que foi decidido não utilizar imagens da mesma câmera em treino e em validação, para evitar qualquer tipo de viés em sua validação. Portanto, cada câmera do conjunto de dados pertence somente ao conjunto de treino ou ao conjunto de validação.

Como as imagens foram capturadas em uma cidade movimentada durante o período

de chuvas, uma variedade de imagens foi utilizada para o treinamento, desde ruas vazias em um dia ensolarado até tráfego intenso em dias chuvosos durante a noite. Devido a chuvas extremamente fortes, algumas lentes de câmeras estavam muito molhadas para distinguir o que era exibido. Essas imagens não foram incluídas no conjunto de dados.

Ao longo do ano, foram captadas imagens de 472 câmeras distintas, com diferentes proporções de imagens de cada classe para cada câmera. Dessa forma, para criar um conjunto de dados balanceado, foram escolhidas as câmeras com quantidade suficiente de imagens de ambas as classes para que todas as câmeras do conjunto de dados tenham a mesma quantidade de imagens.

### 4.1.1 Captação de imagens

As imagens foram captadas ao longo de 2023 em diferentes condições de iluminação e em diferentes épocas de chuva, com variadas intensidades pluviométricas. Foi necessário captar tais imagens de maneira eficiente e inteligente, visto que o sistema de câmeras do COR é composto por milhares de câmeras, tornando custoso o monitoramento de todas elas, além de as chuvas, apesar de intensas, serem imprevisíveis — podendo ocorrer a qualquer hora e com durações bastante variadas.

O sistema de captação foi desenvolvido em Python [86], utilizando a API do Waze para receber notificações de possíveis situações de alagamento. A biblioteca OpenCV [9] foi utilizada para a captura das imagens, e a API do Google Cloud, para upload e armazenamento. Foram utilizados alertas do aplicativo Waze sobre alagamento de ruas e notificações do próprio sistema do COR para definir quais ruas da cidade do Rio de Janeiro estavam alagadas e, assim, iniciar a captação de imagens dessas ruas.

### 4.1.2 Análise de imagens

Para a classificação das imagens, foi criada uma aplicação web baseada em Flask e utilizando MongoDB como banco de dados. As imagens já estavam separadas em seis (6) níveis definidos pelo próprio COR. Estes níveis, em ordem crescente de severidade, são: Normal; Poça; Lâmina; Alagamento; Transbordo; e Bolsão.

Para simplificar, tendo em vista que não há um limite claramente definido entre cada uma dessas classificações, as classes 'Normal' e 'Poça' foram definidas como classe 0, visto que não há impacto no trânsito de veículos. A partir do surgimento de lâminas d'água, o trânsito de veículos começa a ser afetado, visto que estes devem diminuir sua velocidade

para evitar derrapagem [55]. Então, a partir do nível 3, 'Lâmina', as classes mais severas foram definidas como classe 1.

A Figura 5 mostra uma importante avenida na Zona Sul do Rio de Janeiro, no entorno da Lagoa Rodrigo de Freitas: a Av. Epitácio Pessoa, na altura da Rua Maria Quitéria, no lado desta lagoa próximo ao bairro de Ipanema. A Figura 6 mostra a Rua Jardim Botânico, no cruzamento com a Rua Pacheco Leão, no bairro Jardim Botânico. Estas figuras são exemplos de vias classificadas como **não alagadas**, ou classe 0. Quando há água na via, mas ela não impede o tráfego de veículos, decidiu-se que tais situações não seriam classificadas como *alagamento*.


Figura 5: Via completamente seca, rotulada como classe 0

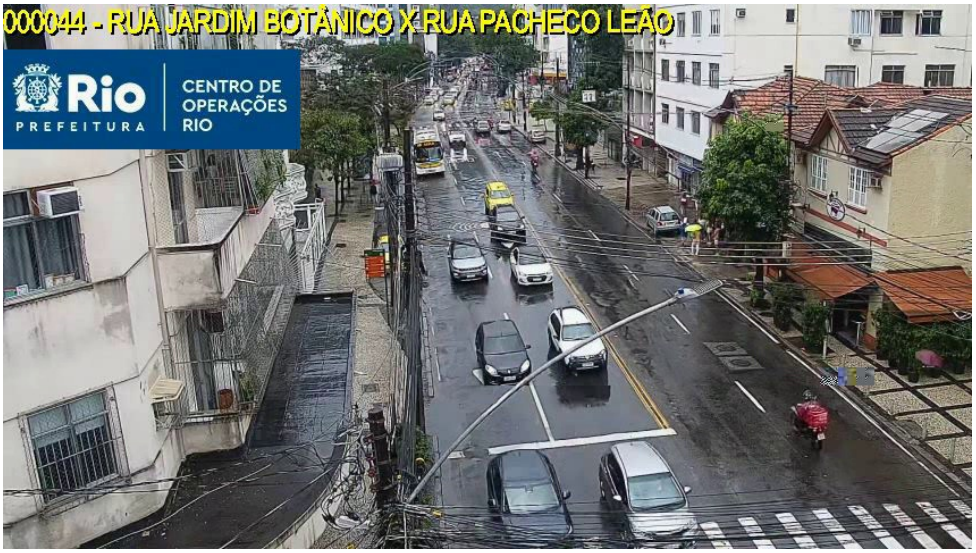


Figura 6: Rua molhada, ainda rotulada como classe 0

Por outro lado, a Fig. 7 e a Fig. 8 mostram capturas de vias classificadas como **ala-**

**gadas**, ou classe 1. Em ambas as imagens, é possível observar água cobrindo grande parte da via, dificultando ou até impossibilitando a passagem de veículos. A Fig. 7 apresenta o exemplo de uma rua localizada na zona norte da cidade, parcialmente alagada, onde o trânsito de veículos é dificultado, embora as calçadas ainda possam ser identificadas. Por sua vez, a Fig. 8 mostra o exemplo de uma rua na Zona Sul, onde não há distinção visual entre calçada e via, ambas completamente submersas. Essas situações constituem a classe *alagamento*.


Figura 7: Rua parcialmente alagada (classe 1).


Figura 8: Avenida totalmente alagada (classe 1).

As Figuras 9 e 10 apresentam, respectivamente, a quantidade de imagens da classe 0 e da classe 1 (eixo vertical) para cada câmera (eixo horizontal). A Figura 11 exibe a distribuição total de imagens por câmera, considerando ambas as classes conjuntamente.

Devido à grande quantidade de câmeras, seus identificadores (ou códigos) não são exibidos no eixo horizontal dessas figuras. Entretanto, a Tabela 10 apresentada no Apêndice A relaciona o identificador de cada câmera e sua respectiva quantidade de imagens para cada classe.

Essas figuras representam o conjunto de dados original, composto por imagens capturadas de 472 câmeras do sistema do COR, sem qualquer balanceamento prévio entre as classes. Nas figuras, a correspondência entre as classes (**alagadas** ou **não alagadas**) e as cores (azul e vermelho) é utilizada para distinguir visualmente a quantidade de imagens de cada classe associada a cada câmera, conforme as barras representadas no eixo horizontal dos histogramas de distribuição.

Observa-se que algumas câmeras possuem significativamente mais imagens da classe 1 (alagadas) do que outras, possivelmente em decorrência da maior incidência de chuvas nas regiões onde essas câmeras estão instaladas. Dessa forma, a utilização desse conjunto de dados, sem uma análise criteriosa da representatividade de cada câmera, poderia introduzir viés no treinamento dos modelos, comprometendo a capacidade de generalização dos mesmos.

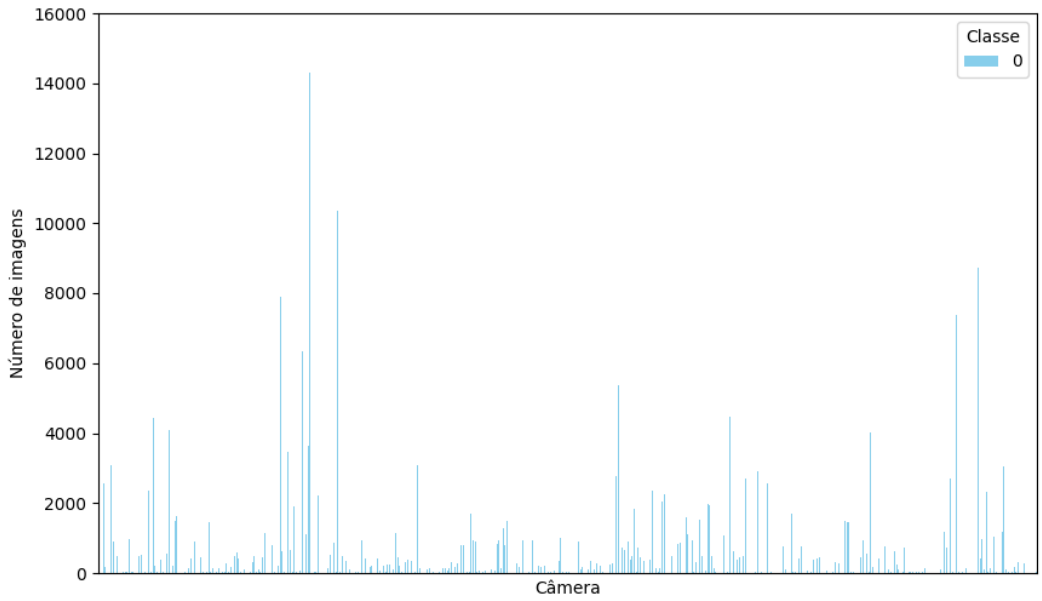


Figura 9: Quantidade de imagens de normalidade por câmera disponível.

A Figura 12 mostra uma distribuição de frequência de câmeras por quantidade de imagens, separadas em intervalos de 100 imagens. O intervalo com maior ocorrência de

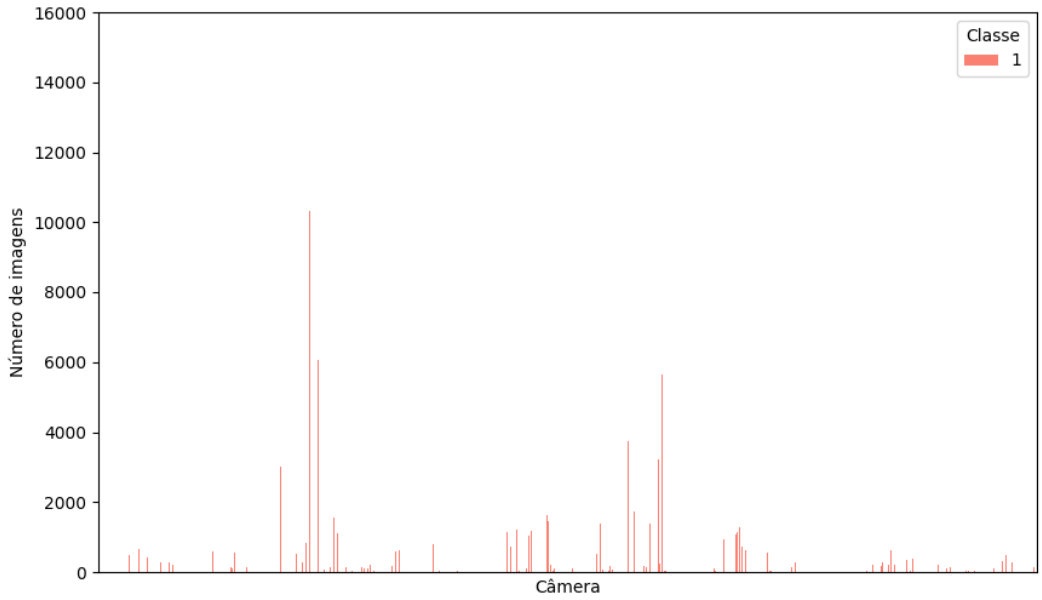


Figura 10: Quantidade de imagens de alagamento por câmera disponível.

câmeras indica o desbalanceamento do conjunto de imagens captadas onde 206 câmeras, ou 43,6%, possuem até 100 imagens ao todo.

A Figura 12 apresenta a distribuição de frequência das câmeras em função da quantidade total de imagens, agrupadas em intervalos de 100 imagens. Observa-se que o intervalo com maior frequência concentra as câmeras que possuem até 100 imagens no total, independentemente da classe. Este dado evidencia de forma clara o desbalanceamento do conjunto de dados, uma vez que 206 câmeras (o que representa aproximadamente 43,6% do total) encontram-se nesse intervalo.

Tal distribuição reforça a necessidade de um criterioso processo de seleção e balanceamento das câmeras que compõem o conjunto final utilizado para treinamento e validação dos modelos. Caso contrário, o treinamento sobre este conjunto poderia sofrer impacto negativo, levando à criação de modelos enviesados, com baixa capacidade de generalização, sobretudo para situações observadas em câmeras com menor representatividade no conjunto de dados original.

Portanto, as arquiteturas de redes neurais convolucionais foram treinadas utilizando um conjunto de dados composto por 77 câmeras, totalizando 4620 imagens. Esse conjunto foi cuidadosamente balanceado, contendo 30 imagens de cada classe para cada câmera — ou seja, 60 imagens por câmera. A divisão dos dados considerou 60 câmeras destinadas

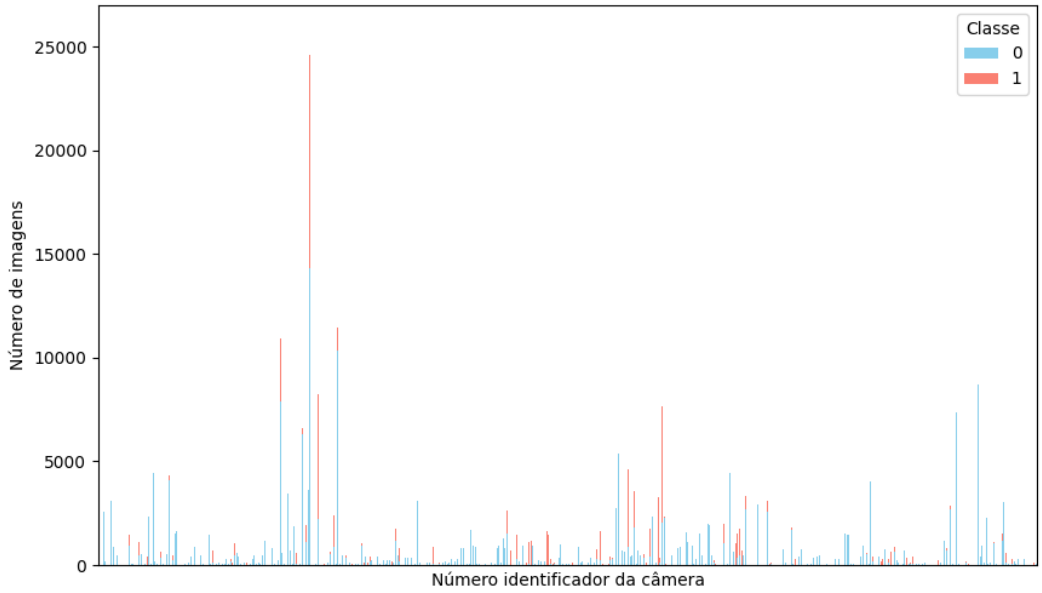


Figura 11: Quantidade de imagens de cada classe por câmera disponível.

ao treinamento e 17 câmeras para validação, correspondendo, respectivamente, a aproximadamente 78% e 22% do total de câmeras.

Adicionalmente, o conjunto de dados excedente — composto pelas câmeras e imagens que não foram selecionadas para os conjuntos de treinamento e validação — foi utilizado para a criação de um conjunto de teste. Esse conjunto de teste foi construído por meio da seleção aleatória de 150 imagens de cada classe, totalizando 300 imagens. Diferentemente do critério adotado para treinamento e validação, na formação do conjunto de teste não houve restrição quanto à representação equitativa de cada câmera, visto que seu objetivo principal é avaliar a capacidade de generalização dos modelos para dados não vistos e com maior variabilidade.

Com isso, o conjunto de dados completo utilizado neste trabalho é composto por 4920 imagens, sendo 4620 destinadas ao treinamento e validação e 300 reservadas para teste, disponível em 10.5281/zenodo.15670835.

Por fim, após a primeira etapa de avaliação dos modelos baseados em *CNN*s, foi realizada uma reanálise da quantidade de imagens por câmera, visando avaliar se diferentes quantidades de imagens poderiam impactar significativamente os índices de desempenho dos modelos — tais como acurácia, precisão, revocação e F1-Score. Esse estudo adicional é detalhado na Seção 5.2.5, onde são discutidos os impactos da variação na quantidade

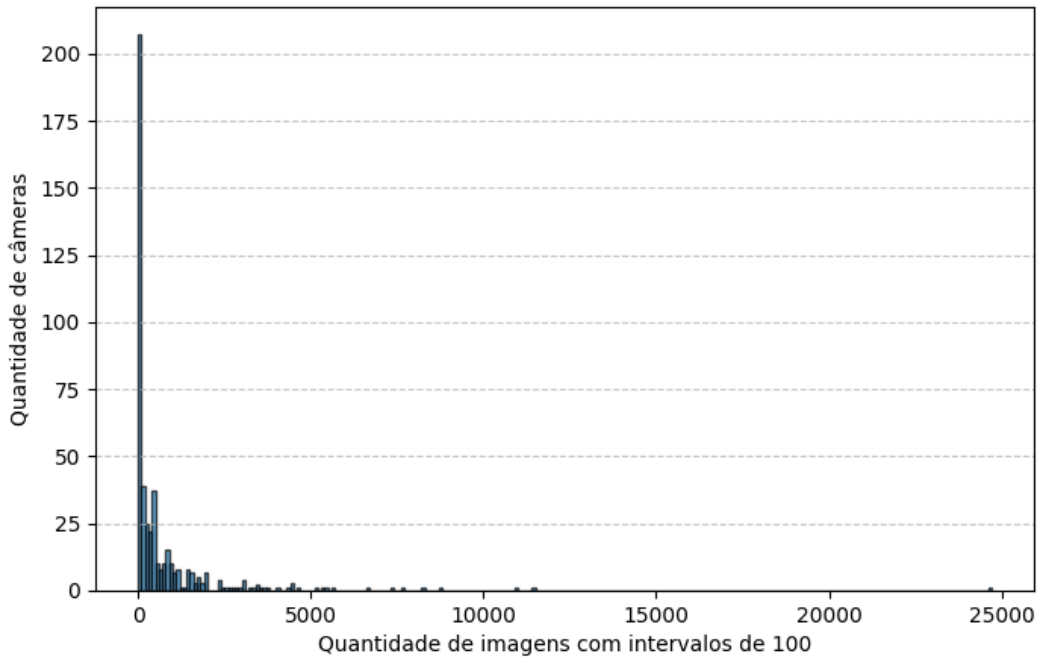


Figura 12: Histograma de quantidade de imagens de alagamento por câmera disponível.

de imagens por câmera sobre o desempenho dos modelos.

### 4.1.3 Pré Processamento

Diversas etapas de pré-processamento foram aplicadas com o objetivo de melhorar a qualidade das imagens e, consequentemente, a capacidade de generalização dos modelos de aprendizado de máquina.

Inicialmente, uma máscara foi aplicada sobre todas as imagens do conjunto de dados para remover o logotipo institucional do COR, localizado na região superior esquerda. Este logotipo consiste no brasão da cidade do Rio de Janeiro, acompanhado de texto em branco sobre um fundo azul. A remoção dessa informação visual foi considerada essencial, uma vez que sua presença constante poderia introduzir um viés indesejado no processo de aprendizado, levando o modelo a associar características do logotipo às classes de alagamento, comprometendo sua capacidade de generalização.

A estratégia adotada para essa remoção consistiu na aplicação de uma máscara de preenchimento nulo (valores de pixel zerados) sobre a região específica ocupada pelo logotipo. Tal abordagem foi viável devido à posição fixa e invariante desse elemento em todas as imagens. No entanto, não foi possível realizar uma reconstrução semântica ou

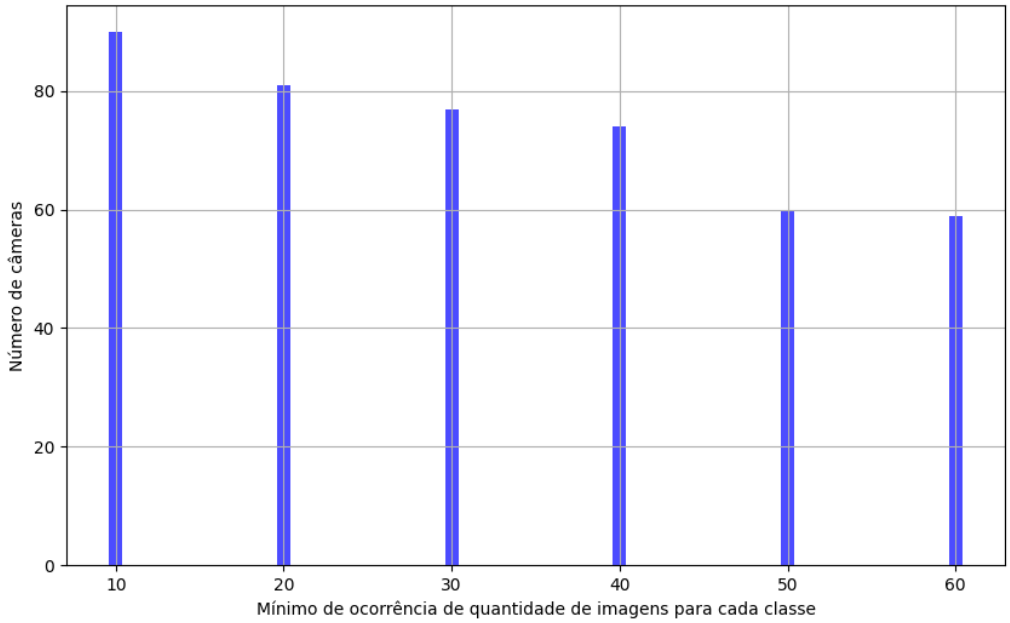


Figura 13: Quantidade de câmeras com um número mínimo de imagens de cada classe.

interpolação da região ocultada, uma vez que não existiam quadros de referência sem a presença do logotipo no dataset original.

Em uma segunda etapa, buscou-se corrigir problemas relacionados à exposição excessiva presentes em parte das imagens, especialmente aquelas capturadas sob condições de elevada luminosidade. Para isso, foi realizada a conversão do espaço de cor original, do modelo RGB para o espaço de cor $YC_rC_b$. Esta conversão foi escolhida porque isola o componente de luminância (canal $Y$) dos componentes cromáticos ($C_r$ e $C_b$), permitindo intervenções específicas na intensidade da claridade da imagem sem afetar suas características de cor.

O ajuste da claridade foi realizado através de uma transformação linear aplicada exclusivamente sobre o canal de luminância ($Y$), segundo a seguinte equação:

$$Y' = FC \cdot Y \tag{4.1}$$

onde $FC$ representa o Fator de Claridade, um escalar real positivo no intervalo $(0, 1]$. Este fator controla a intensidade da redução da luminosidade. Na fase inicial dos experimentos, adotou-se $FC = 0{,}5$ como valor de referência, com o objetivo de reduzir a claridade à metade. Valores distintos para $FC$ foram posteriormente analisados na Seção

5.2, buscando otimizar o desempenho do modelo.

Após a aplicação do fator de claridade, as imagens foram reconvertidas do espaço $YC_rC_b$ para RGB, garantindo que pudessem ser visualizadas e processadas de acordo com os padrões dos modelos de *CNN* utilizados.

As Figuras 14 e 15 ilustram o efeito visual da aplicação do fator de claridade. A primeira apresenta a imagem na condição original, enquanto a segunda demonstra a mesma cena após a aplicação de $FC = 0{,}5$. É possível observar uma significativa redução na intensidade de claridade, preservando, contudo, a integridade das informações cromáticas e estruturais relevantes para a tarefa de classificação.


Figura 14: Exemplo de região sem alteração no canal $Y$.

Para introduzir mais variedade às imagens no treinamento, em seguida foi aplicada uma inversão horizontal aleatória com uma probabilidade de 50% nos dados de entrada. Em outras palavras, cada imagem da divisão de treinamento tem 50% de chance de ser invertida horizontalmente ou não.

Além disso, transformações padrão de redimensionamento, corte e normalização de intensidades são usadas para preparar as imagens para cada modelo *CNN*. É aplicado aos


Figura 15: Exemplo de mesma região com 50% do valor original de $Y$.

dados, inicialmente, um processo de redimensionamento para transformar cada quadro, ajustando-o ao tamanho esperado de cada modelo, onde as imagens com dimensão 854 x 480 pixels são redimensionadas para 256 x 256 pixels. Então, elas são submetidas a um corte de dimensão 224 x 224 pixels, com o mesmo centro da imagem. Com exceção do *Inception*, que redimensiona para 342 x 342 pixels e realiza um corte central de 299 x 299 pixels. Os valores do conteúdo de cada pixel são redimensionados para estarem no intervalo [0,1].

Finalmente, todo o conjunto de dados foi normalizado pela média e desvio padrão específicos de cada modelo: todos os modelos avaliados têm a mesma matriz média de (0,485; 0,456; 0,406) e matriz de desvio padrão de (0,229; 0,224; 0,225) para cada canal $RGB$, usados para normalizar a imagem.

## 4.2 Modelos de classificação

Com base na nos trabalhos da literatura sobre alagamento urbano anteriormente apresentados, os modelos MobileNetV2 [71], VGG19 [78], InceptionV3 [80] e DenseNet [40] foram avaliados. Além disso, a arquitetura de *Visual Transformer* ($ViT$)[48] foi incluída. Essa adição foi feita porque vários modelos de classificação de imagens de alto desempenho são baseados em arquiteturas $Visual\ Transformers$, como $ViT - Huge$ usado no conjunto de dados CIFAR-10 [27] e $ViT - Large$ no conjunto de dados STL-10 [31, 46].

Todos os modelos foram treinados em 50 épocas, onde o *Early Stopping* foi empregado com uma restrição de 10 épocas e decaimento da taxa de aprendizagem de 0,1. Três diferentes taxas de aprendizagem foram aplicadas em todos os modelos, visto que as diferentes complexidades dos modelos podem resultar em diferentes taxas de aprendizagem ótimas. O *Stochastic Gradient Descent* ($SGD$) foi usado como otimizador inicialmente, entretanto, outros otimizadores foram testados após a avaliações dos modelos, como pode ser visto na seção 5.3.

A transferência de aprendizado foi aplicada a todos os modelos, inicializando-os com os pesos padrões do IMAGENET. Isso foi feito para utilizar os pesos aprendidos anteriormente para auxiliar no aprendizado dos pesos das novas classes [49]. Posteriormente, a última camada totalmente conectada foi modificada para ter apenas duas saídas, se adequando ao problema de classificação binária.

## 4.3 Avaliação dos modelos

Para avaliar o desempenho dos cinco modelos com diferentes taxas de aprendizagem, inicialmente foi utilizada a acurácia para distinguir o melhor modelo. Após a identificação do melhor modelo, a precisão e a revocação também foram utilizadas para melhor entender seu desempenho.

Como o conjunto de dados está completamente balanceado, conforme mencionado na Seção 4.1, com ambas as classes igualmente representadas, esses avaliadores fornecem um cenário adequado para a compreensão dos resultados. Adicionalmente, todas as câmeras possuem o mesmo número de imagens, garantindo, assim, uma representação equilibrada entre as fontes de dados e suas distribuições pela cidade.

Para calcular a acurácia, a precisão e a revocação, inicialmente foram obtidos os seguintes valores: Verdadeiros Positivos (TP), que correspondem ao número de rótulos

positivos classificados corretamente pelo modelo; Verdadeiros Negativos (TN), que representam os rótulos negativos classificados corretamente; Falsos Positivos (FP), que são os rótulos negativos classificados incorretamente como positivos; e Falsos Negativos (FN), que indicam o número de rótulos positivos classificados de forma incorreta como negativos.

O primeiro índice de avaliação selecionado para análise foi a acurácia, pois ela fornece uma visão geral da capacidade do modelo em classificar corretamente os exemplos em um conjunto de dados balanceado.

A acurácia pode ser calculada pela seguinte fórmula:

$$Acuracia = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.2}$$

O índice de avaliação de precisão indica a porcentagem de imagens classificadas como positivas que realmente pertencem à classe positiva [30]. É especialmente relevante para aumentar a confiabilidade do modelo na identificação correta da situação de inundação.

A Equação 4.3 formaliza o cálculo da precisão:

$$Precisao = \frac{TP}{TP + FP} \tag{4.3}$$

Além disso, a revocação pode ser empregada como um avaliador final para calcular a taxa de imagens positivas classificadas corretamente [57]. Ela é útil para garantir que o menor número possível de situações de inundação seja classificado incorretamente. A revocação mede a capacidade do modelo de identificar corretamente todas as instâncias relevantes. A equação a seguir define o cálculo da revocação:

$$Revocacao = \frac{TP}{TP + FN} \tag{4.4}$$

O próximo capitulo apresenta os resultados obtidos.

# 5 Experimentos e Resultados

Esta seção apresenta os resultados das cinco arquiteturas analisadas. Em seguida, a arquitetura com melhor desempenho é analisada mais detalhadamente sob diferentes aspectos. Toda a pesquisa foi realizada em uma máquina com placa gráfica NVIDIA GeForce RTX 3060 Ti, processador Intel Core i7-11700K de 3,60 GHz e 16 GB de memória RAM.

## 5.1 Resultados dos Modelos

A Tabela 2 exibe a acurácia de cada modelo com diferentes taxas de aprendizagem, utilizando o *SGD* como otimizador. A taxa de aprendizagem de $1x10^{-5}$ apresentou as melhores acurácias no geral, porém não gerou o melhor modelo. Nenhum modelo obteve boa acurácia com essa taxa de aprendizagem, indicando que um valor muito baixo limitou a capacidade dos modelos de convergir para uma solução satisfatória.

O *MobileNetV2* obteve os piores resultados nas outras duas taxas de aprendizagem avaliadas. Já o *ViT* apresentou o melhor desempenho, com acurácia superior tanto para a taxa de aprendizagem de $1x10^{-3}$ quanto para a de $1x10^{-4}$.

O desempenho inferior do *MobileNetV2* pode ser explicado por sua arquitetura, proje-

Tabela 2: Acurácia dos cinco modelos com diferentes taxas de aprendizagem.

| Modelos | Taxa de aprendizagem | | |
|---|---|---|---|
| | 1x10-03 | 1x10-04 | 1x10-05 |
| VGG19 | 0,808 | 0,732 | 0,524 |
| InceptionV3 | 0,795 | 0,731 | 0,529 |
| DenseNet | 0,802 | 0,658 | 0,592 |
| MobileNetV2 | 0,781 | 0,608 | 0,534 |
| ViT | 0,816 | **0,826** | 0,511 |

tada para uso em dispositivos com menor poder computacional [71], possuindo, portanto, um número significativamente menor de parâmetros treináveis em comparação com as demais arquiteturas avaliadas.

Um comportamento comum a todos os modelos, exceto o *ViT*, foi a redução da acurácia conforme a diminuição da taxa de aprendizagem. Isso sugere que taxas de aprendizagem muito baixas podem ser inadequadas para ajustar a grande quantidade de parâmetros presentes em redes profundas.

Apesar de tanto o *VGG* quanto o *DenseNet* terem alcançado acurácias próximas a 80%, o *ViT* apresentou resultados ligeiramente superiores. Outro aspecto relevante a ser considerado é que, enquanto o *InceptionV3* levou 47 minutos para completar seu treinamento, o *ViT* realizou o mesmo em apenas 22 minutos no mesmo conjunto de dados. Portanto, o *ViT* utilizando taxa de aprendizado de $1x10^{-4}$ foi selecionado como o modelo a ser utilizado para as etapas subsequentes deste trabalho.

## 5.2 Avaliação do melhor modelo

Após a escolha da arquitetura de melhor desempenho e da taxa de aprendizado adequada, outras análises devem ser feitas para saber se é possível melhorar seus resultados e o quão adequado foi seu treinamento.

### 5.2.1 Validação cruzada K-Fold

A primeira análise realizada exclusivamente com o *ViT* foi a aplicação de validação cruzada do tipo K-Fold, utilizando o mesmo conjunto de dados. A Tabela 3 apresenta as métricas de acurácia, precisão e revocação obtidas nos cinco subconjuntos, ou *folds*, empregados na validação.

A acurácia média de 0,801 obtida na validação cruzada foi inferior à acurácia de 0,826 observada na validação simples. Esse resultado sugere que o conjunto de treino inicialmente utilizado pode ter incluído imagens mais facilmente classificáveis, o que beneficiou o desempenho na validação simples.

Ao analisar cada subconjunto individualmente, observa-se que os folds 2 e 5 apresentaram acurácias significativamente menores em relação aos demais. Tal comportamento indica que as características das câmeras incluídas nesses subconjuntos de validação não foram adequadamente aprendidas pelo modelo, que foi treinado com imagens provenientes

Tabela 3: Métricas do ViT com validação cruzada K-Fold com K=5.

| Subconjunto | Acurácia | Precisão | Revocação |
|---|---|---|---|
| 1 | 0,833 | 0,838 | 0,827 |
| 2 | 0,777 | 0,773 | 0,785 |
| 3 | 0,794 | 0,780 | 0,820 |
| 4 | 0,819 | 0,801 | 0,849 |
| 5 | 0,771 | 0,816 | 0,700 |
| Média | 0,794 | 0,801 | 0,820 |

das outras câmeras. Além disso, a baixa revocação observada no subconjunto 5 pode indicar que o modelo adotou uma estratégia conservadora, prevendo a classe de normalidade apenas em situações de classificação com maior confiança, ao custo do aumento de falsos negativos.

Embora o modelo tenha mantido uma acurácia média aceitável, o desempenho inferior nos subconjuntos 2 e 5 evidencia uma sensibilidade dos resultados à composição das câmeras no conjunto de treino, ressaltando a importância de uma seleção equilibrada das câmeras para garantir generalização adequada do modelo.

### 5.2.2 Métricas no conjunto de teste

Após a validação cruzada, o modelo também foi avaliado no conjunto de teste, o qual apresenta balanceamento entre as classes, porém não possui balanceamento na quantidade de imagens provenientes de cada câmera que compõe esse conjunto.

A Tabela 4 exibe os resultados obtidos pelo modelo no conjunto de teste, juntamente com seus resultados na validação cruzada. Observa-se uma leve queda na acurácia, contudo o modelo mantém uma capacidade razoável de classificação. O aumento na precisão indica que o modelo cometeu menos falsos positivos no conjunto de teste, evidenciando uma classificação mais conservadora da classe positiva.

Entretanto, houve uma queda significativa na revocação no conjunto de teste. Dessa forma, apesar da postura conservadora, o modelo falhou em classificar corretamente uma quantidade expressiva de verdadeiros positivos.

Tabela 4: Métricas do ViT no conjunto de teste.

| Conjunto | Acurácia | Precisão | Revocação |
|---|---|---|---|
| Validação Cruzada | 0,794 | 0,801 | 0,820 |
| Teste | 0,780 | 0,814 | 0,719 |

Tabela 5: Redução de áreas de brancos e desempenho do $ViT$

| Fatores de claridade | Acurácia | Precisão | Revocação |
|---|---|---|---|
| 0,25 | 0,703 | 0,758 | 0,604 |
| 0,5 | **0,826** | **0,863** | **0,778** |
| 0,75 | 0,769 | 0,802 | 0,738 |
| 1 | 0,756 | 0,782 | 0,708 |

### 5.2.3 Diminuição da intensidade luminosa

Após a escolha da arquitetura de melhor desempenho, diferentes valores para o fator denominado nesta dissertação como fator de claridade ($FC$), conforme mencionado na Seção 4.1.3, foram avaliados utilizando o $ViT$, com o objetivo de analisar se o modelo poderia ser aprimorado ao levar em consideração a claridade excessiva presente em algumas imagens.

A Tabela 5 apresenta o desempenho do $ViT$ com quatro valores distintos para o $FC$, sendo que o valor 1 para o $FC$ indica que a intensidade luminosa da imagem não foi alterada, ou seja, a quantidade de regiões com pixels na cor branca permanece inalterada em relação à imagem original. A redução do $FC$ promoveu uma melhora gradual no desempenho do modelo, entretanto, um $FC$ muito baixo impactou negativamente o resultado, como evidenciado pelos resultados obtidos com $FC$ igual a 0,25. A diminuição do $FC$ para 0,75 gerou uma melhora marginal nos resultados, contudo a escolha inicial de 0,5 mostrou-se a mais adequada dentre os valores testados.

### 5.2.4 Curvas de acurácia e perda do melhor modelo

A Figura 16 e a Figura 17 apresentam o desempenho do $ViT$ durante as fases de treinamento (linha azul) e validação (linha vermelha).

No treinamento, a acurácia (Figura 16) converge para aproximadamente 0,87, enquanto na validação mantém-se em torno de 0,83. Entretanto, ao observar apenas a

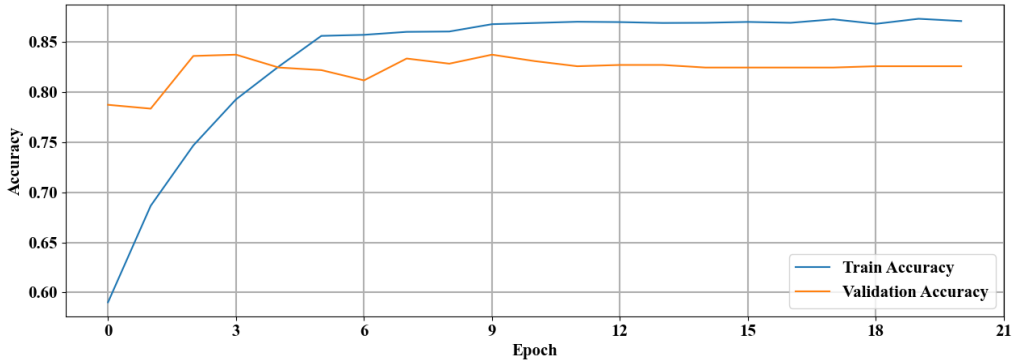


Figura 16: Acurácia do $ViT$ durante treino e validação.

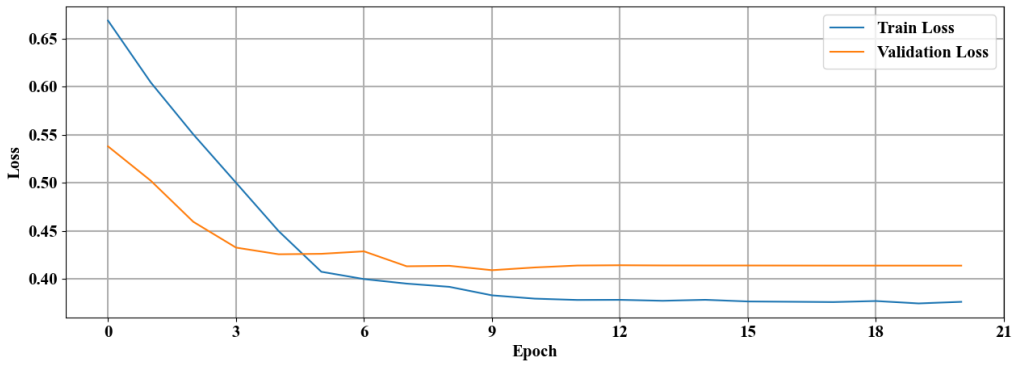


Figura 17: Perda do $ViT$ durante treino e validação.

acurácia no conjunto de validação, nota-se que ela iniciou com valores relativamente altos antes de convergir. A pequena discrepância entre as acurácias de treinamento e validação indica que não houve $overfitting$ dos dados, demonstrando que o modelo foi capaz de generalizar adequadamente.

Ao analisar a perda ($loss$) durante o treinamento (Figura 17), observa-se que, apesar da alta acurácia no conjunto de treino, a perda permaneceu relativamente elevada inicialmente, estabilizando-se em torno de 0,37 (linha azul), enquanto a perda no conjunto de validação estabilizou-se em aproximadamente 0,43 (linha vermelha).

A pequena diferença na acurácia entre os conjuntos de treinamento e validação (Fig. 16) pode ser justificada pela introdução de características no conjunto de validação que não estão presentes no conjunto de treinamento.

### 5.2.5 Melhor modelo com diferentes quantidades de câmeras

Com o objetivo de verificar se o aumento da quantidade de câmeras no conjunto de treino levaria a melhorias significativas em sua performance, a arquitetura *ViT* foi treinada com conjuntos de treino menores. Esses conjuntos possuem menor quantidade de câmeras, mas mantêm o mesmo número de imagens por classe para cada câmera.

Para compor esses conjuntos comparativos ao utilizado originalmente, dentre as 60 câmeras que compõem o conjunto original, foram escolhidas aleatoriamente 30 câmeras para formar o segundo conjunto. Dessas 30 câmeras, foram selecionadas aleatoriamente 15 para compor o terceiro conjunto. Portanto, os três conjuntos de treino possuem as mesmas 15 câmeras, e o conjunto com 60 câmeras inclui todo o conjunto das 30 câmeras, acrescido de outras 30 câmeras adicionais.

Dessa forma, o efeito de adicionar novas câmeras ao conjunto de treino pode ser mais facilmente compreendido. A Tabela 6 apresenta a relação entre o tamanho do conjunto de treino, sua acurácia e o tempo de treinamento. O modelo foi testado utilizando o mesmo conjunto de teste pertencente ao conjunto de dados original.

Tabela 6: Acurácia × tempo de treino × número de imagens usadas

| **Nº de Câmeras** | **Acurácia** | **Tempo (minutos)** |
|---|---|---|
| 15 | 0,728 | 17 |
| 30 | 0,804 | 18 |
| 60 | 0,826 | 21 |

Como esperado, aumentar a diversidade de câmeras no conjunto de treino e, consequentemente, a quantidade total de imagens de treino, resultou em incremento na acurácia do modelo. Entretanto, o número de câmeras atualmente utilizado mostra-se adequado, pois a inclusão de mais câmeras não trouxe melhorias significativas no desempenho. Observou-se que dobrar a quantidade de imagens de treino, passando de 1800 para 3600 — ou seja, de 30 para 60 câmeras no conjunto de treino — representou uma melhoria marginal de apenas 0,022 na acurácia.

## 5.3 Comparação entre diferentes otimizadores

Apesar de, inicialmente, todos os modelos terem sido treinados utilizando o *SGD* como otimizador, a escolha de diferentes otimizadores e taxas de aprendizagem pode influenciar

Tabela 7: Acurácia do *ViT* com diferentes otimizadores.

| Taxa de Aprendizado | Acurácia SGD | Adam | NAdam |
|---|---|---|---|
| 1x10-06 | 0,504 | 0,833 | 0,819 |
| 1x10-05 | 0,512 | 0,841 | **0,862** |
| 1x10-04 | 0,826 | 0,636 | 0,650 |

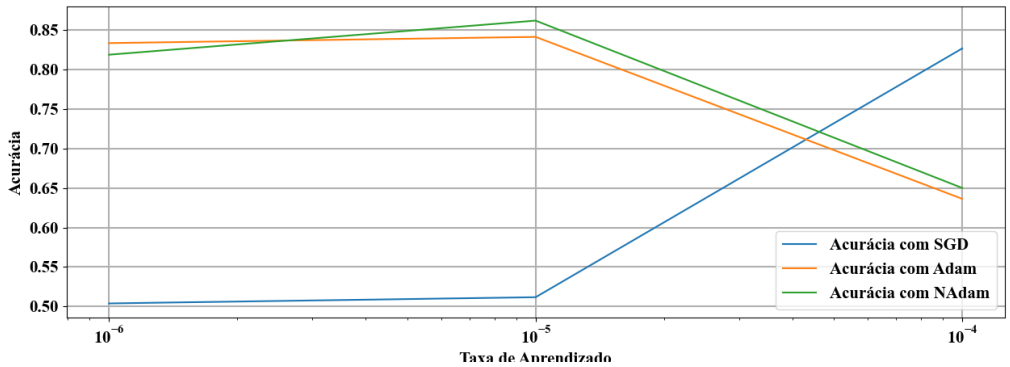


Figura 18: Acurácia do *ViT* com diferentes otimizadores.

significativamente a performance das *CNN*.

Com base nos resultados apresentados por Dogo *et al.* [26], que compararam diversos otimizadores em redes neurais e conjuntos de dados de diferentes tamanhos e complexidades, esta dissertação realizou uma comparação dos resultados do *ViT* utilizando o *SGD*, o *Adaptive Moment Estimation* [47] e o *Nesterov-accelerated Adaptive Moment Estimation* [28]. Todos os otimizadores foram configurados conforme descrito na Seção 4.2, sendo testados com diferentes taxas de aprendizagem iniciais.

Os resultados apresentados na Tabela 7 indicam que tanto o *ADAM* quanto o *NADAM* apresentaram acurácias superiores ao *SGD*. O *NADAM* obteve a melhor acurácia, alcançando 0,86 com uma taxa de aprendizado de $1x10^{-5}$.

Ao analisar o desempenho do *SGD*, observa-se que taxas de aprendizado menores resultaram em piores resultados, possivelmente indicando que o passo de atualização foi muito pequeno para escapar de mínimos locais, especialmente em arquiteturas mais complexas, como o *ViT*.

Nenhum dos otimizadores conseguiu obter um bom desempenho com a menor taxa de aprendizado, o que pode ser explicado pelo fato de que o modelo não realizou atualizações significativas nos pesos durante o treinamento. As variantes do *ADAM* apresentaram

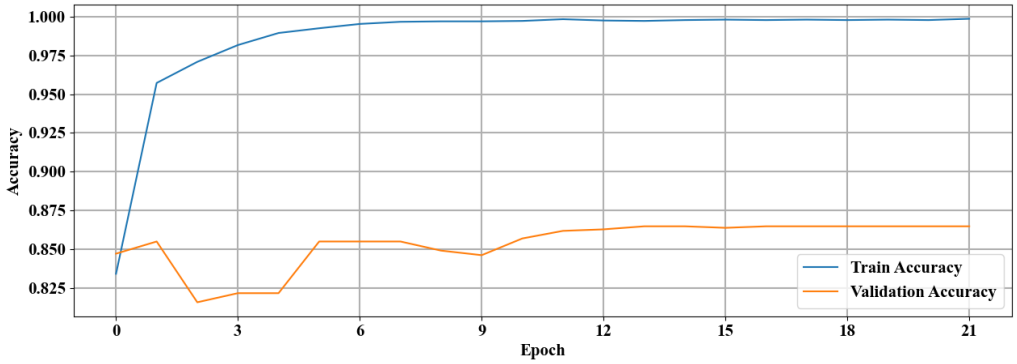


Figura 19: Acurácia do *ViT* com NAdam.

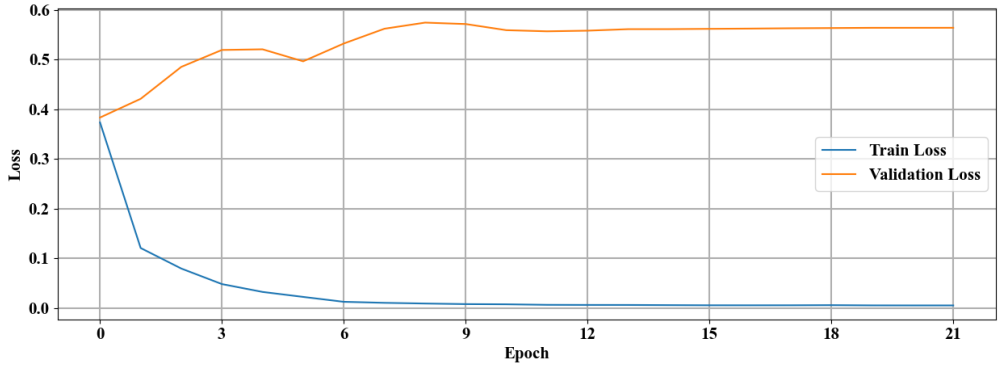


Figura 20: Perda do *ViT* com NAdam.

melhor performance com taxas de aprendizado moderadas, com o *NADAM* se destacando. A utilização da aceleração de Nesterov para atualizações mais rápidas no *NADAM* pode ser a principal responsável por seu desempenho superior.

As Figuras 19 e 20 ilustram a acurácia e a perda do *ViT* durante o treinamento com o *NADAM*, utilizando a taxa de aprendizado de $1x10^{-5}$, momento em que foi obtida a maior acurácia de validação. Considerando que o modelo estabilizou sua acurácia de treinamento em 1 , enquanto a acurácia de validação se manteve em torno de 0,862, observa-se que o modelo não conseguiu generalizar adequadamente, indicando a ocorrência de *overfitting*. Esse fenômeno é corroborado pela curva de perda, na qual a perda de validação aumentou, em contraste com a perda de treinamento, que diminuiu para valores próximos a zero.

Fazendo a mesma análise para a taxa de aprendizagem de $1x10^{-6}$, onde a acurácia de validação foi um pouco menor, pode-se notar algumas semelhanças e diferenças. Ainda houve uma diferença significativa entre as acurácias de treino e validação, com a acurácia de treino estabilizando em torno de 0,984, enquanto a acurácia de validação ficou em aproximadamente 0,832.

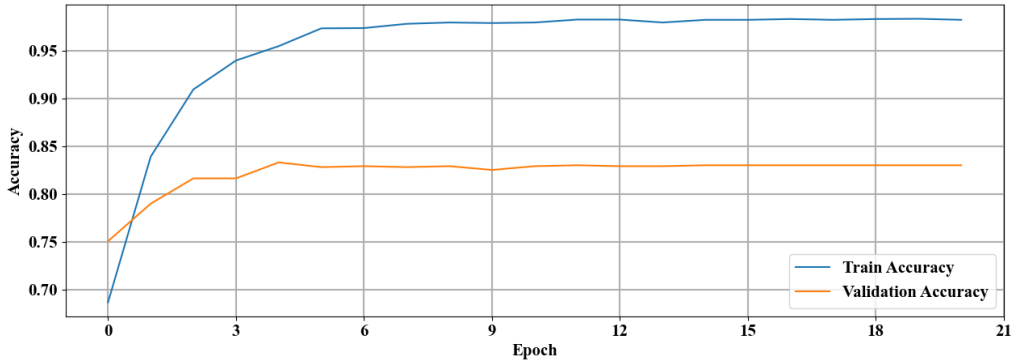


Figura 21: Acurácia do *ViT* com NAdam e menor taxa de aprendizagem.

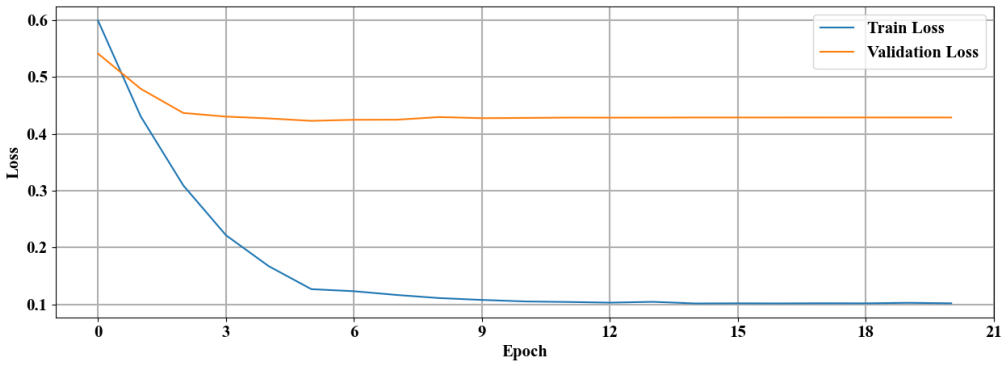


Figura 22: Perda do *ViT* com NAdam e menor taxa de aprendizagem.

Entretanto, a curva de perda apresentou melhorias. Tanto a curva de treino quanto a curva de validação tenderam a diminuir, embora a redução da perda na validação não tenha sido tão significativa. Este modelo não apresentou sinais claros de *overfitting* como o modelo com maior taxa de aprendizagem, mas ainda demonstra potencial para melhorias.

Visto que o modelo *ViT* usando *NADAM* e taxa de aprendizagem de $1x10^{-6}$ obteve acurácia de validação semelhante a do modelo com o *SGD* de taxa de aprendizagem de $1x10^{-4}$, entretanto, teve maior diferença entre as acurácias de treino e validação, assim como redução na função de perda, o modelo com *SGD* vai continuar a ser analisado como o melhor modelo, por possuir maior estabilidade.

## 5.4 Comparações entre cena diurna e noturna

Para melhor compreender o efeito da iluminação da cena nos resultados de classificação do modelo [63], o melhor desempenho do *ViT* com o otimizador *SGD*, descrito na Seção 5.3, foi avaliado separadamente em relação às suas métricas para imagens capturadas durante

Tabela 8: Métricas do *ViT* de dia e à noite.

| Período do dia | Acurácia | Precisão | Revocação |
|---|---|---|---|
| Diurno | 0,802 | 0,702 | 0,694 |
| Noturno | 0,758 | 0,800 | 0,758 |

Tabela 9: Performance do *ViT* em outros conjuntos de dados

| Conjunto de Dados | Acurácia | Precisão | Revocação |
|---|---|---|---|
| *RFD* | 0,826 | 0,863 | 0,778 |
| EFD | 0,731 | 0,892 | 0,714 |
| Sazara | 0,744 | 0,903 | 0,674 |

o dia e durante a noite no local. Os resultados dessa avaliação estão apresentados na Tabela 8.

Em contraste com os resultados apresentados e discutidos no Capítulo 3, nos quais o modelo exibiu desempenho substancialmente superior em imagens diurnas quando comparado às imagens noturnas [63], o modelo treinado com o conjunto de dados *RFD* demonstrou diferenças menores na acurácia entre imagens capturadas em condições diurnas e noturnas. Essa maior estabilidade na acurácia pode ser atribuída à redução da claridade das imagens, obtida por meio do pré-processamento aplicado.

## 5.5 Outros conjuntos de dados

O modelo *ViT* treinado no conjunto *RFD* também foi avaliado em dois outros conjuntos de dados, o European Flood Dataset (*EFD*) [6] e o conjunto de dados de Sazara *et al.* [72]. O *EFD* é composto por 327 imagens da classe normalidade e 252 imagens da classe alagamento, totalizando 579 imagens. Já o conjunto de dados de Sazara *et al.* contém 491 imagens, das quais 238 correspondem à classe normalidade e 253 à classe alagamento.

Observa-se, na Tabela 9, que o desempenho do modelo diminuiu ao ser testado nestes conjuntos distintos. Apesar do aumento na precisão, houve uma queda significativa na acurácia, indicando que o modelo não conseguiu generalizar adequadamente para imagens com diferentes ângulos, qualidade e condições de iluminação.

# 6 Conclusão

Com o objetivo de criar um modelo de classificação para classificar o estado de alagamento das ruas da cidade do Rio de Janeiro, esta pesquisa organizou um conjunto de dados original ao longo de meses de captação e classificação de imagens do sistema de câmeras do COR, denominado *RFD*, disponibilizado em 10.5281/zenodo.15670835. Com este conjunto de dados, foi realizada uma comparação entre as arquiteturas *VGG*, *Inception*, *DenseNet*, *MobileNet* e *ViT* no problema de detecção de alagamentos, para achar o melhor modelo entre as arquiteturas selecionadas para alcançar o objetivo do trabalho.

Os resultados mostraram que o *ViT* superou as outras arquiteturas com diferentes taxas de aprendizagem. A alteração da intensidade luminosa da imagem teve impacto claro no desempenho do modelo e ajudou a diminuir a diferença de acurácia entre as imagens noturnas e diurnas, em comparação aos resultados descritos em Piedad *et al.* [63].

O modelo *ViT* se destacou usando *SGD* como otimizador e com taxa de aprendizagem de $1e^{-4}$. Foi o melhor não só pelas suas métricas, mas pela sua maior estabilidade na curva de perda e pela menor discrepância entre acurácia de treino e validação.

Todos os modelos foram treinados utilizando imagens capturadas pelo sistema de câmeras do COR, onde o *ViT* obteve acurácia de 83%. Este trabalho também analisou o modelo obtido em outras fontes de dados; o modelo conseguiu acurácia de 73% no conjunto de dados *EFD* [6], e de 74% no conjunto de dados de Sazara *et al.* [72].

Ao abordar o complexo problema de classificar imagens de alagamento de uma cidade grande e diversa como o Rio de Janeiro, este trabalho conseguiu alcançar seu objetivo de concluir uma abordagem inicial para a classificação automática da situação de alagamento de ruas, através de um conjunto de dados organizado e disponibilizado por esta dissertação, baseado na própria cidade do Rio de Janeiro (RJ).

## 6.1 Trabalhos futuros

Nesta dissertação, foi utilizado em todo o conjunto de treino o fator de claridade para diminuir o efeito de claridade excessiva em algumas imagens. Buscar uma abordagem que afete somente as imagens que possuam esse problema, e até mesmo somente na região com tal claridade.

A integração de diferentes conjuntos de dados ao treinamento do modelo pode impactar positivamente seu desempenho. Embora diferentes quantidades de tráfego tenham sido utilizadas no treinamento, outros eventos que podem ocorrer na rua ou ao seu redor, como manutenção das vias ou construções, não foram incluídos no conjunto de dados de treinamento. Portanto, essas situações poderiam ser classificadas incorretamente pelo modelo, e imagens desses eventos deveriam ser incluídas no conjunto de dados de treinamento.

Apesar do otimizador *SGD* inicialmente escolhido apresentar melhor desempenho, os resultados com o *NADAM* mostram que mais estudos sobre a escolha de diferentes valores para os parâmetros do otimizador podem aprimorar o modelo atual.

Além disso, alterações na arquitetura, como a criação de novas camadas demonstrada em Agung *et al.* [1], podem melhorar o desempenho do modelo, uma vez que eles alcançaram uma precisão de 0,95 para o *MobileNetV2*, enquanto este trabalho alcançou uma precisão mais baixa de 0,78 com a mesma arquitetura.

O uso de outros instrumentos meteorológicos de medição, como pluviômetros instalados por toda a cidade, pode ser usado para corroborar as classificações resultantes do modelo gerado. Modelos baseados em imagens de satélites seriam importantes para uma maior eficácia de previsões. Outros aspectos importantes a serem incluídos são dados ligados ao histórico de regiões de alagamentos e à própria geografia da cidade, onde lugares próximos de encostas e mais baixos são as regiões que sabidamente devem ser mais monitoradas. No entanto, como o escopo da pesquisa foi o desenvolvimento de um modelo de classificação e algumas ruas não possuem sensores de nível de água, esse tipo de análise foi deixado de lado para uma abordagem inicial.

# REFERÊNCIAS


1 AGUNG, Andi Sadri et al. Urban Flood Disaster Mitigation through Image Classification Using Transfer Learning Method with MobileNet Fine-tuning. In: 2023 International Conference on Advanced Mechatronics, Intelligent Manufacture and Industrial Automation (ICAMIMIA). [S. l.: s. n.], 2023. Pp. 364–369. DOI: 10.1109/ICAMIMIA60881.2023.10427560.

2 AHMED, Imran et al. An Internet of Things and AI-Powered Framework for Long-Term Flood Risk Evaluation. **IEEE Internet of Things Journal**, v. 11, n. 3, pp. 3812–3819, 2024. DOI: 10.1109/JIOT.2023.3308564.

3 AKSHYA, J.; PRIYADARSINI, P.L.K. A Hybrid Machine Learning Approach for Classifying Aerial Images of Flood-Hit Areas. In: 2019 International Conference on Computational Intelligence in Data Science (ICCIDS). [S. l.: s. n.], 2019. Pp. 1–5. DOI: 10.1109/ICCIDS.2019.8862138.

4 ALHADY, M. Athallah Dzikri et al. Comparative Performance of Water Index for Water Segmentation Model Using U-Net Architecture. In: 2024 International Conference on Computer, Control, Informatics and its Applications (IC3INA). [S. l.: s. n.], 2024. Pp. 84–88. DOI: 10.1109/IC3INA64086.2024.10732848.

5 ARORA, Gautam; R M, Bhavadharini. Disaster Scene Classification with Deep Learning: A Keras-Based Approach Utilizing Robotic Systems. In: 2024 IEEE Students Conference on Engineering and Systems (SCES). [S. l.: s. n.], 2024. Pp. 1–6. DOI: 10.1109/SCES61914.2024.10652476.

6 BARZ, Björn et al. Enhancing Flood Impact Analysis using Interactive Retrieval of Social Media Images. Versão inglesa. **Archives of Data Science, Series A (Online First)**, v. 5, n. 1, a06, 21 s. online, 2018. ISSN 2363-9881. DOI: 10.5445/KSP/1000087327/06.

7 BISHOP, Christopher. **Pattern Recognition and Machine Learning**. [S. l.]: Springer, 2007. ISBN 9780387310732.

8 BOUCHELKIA, Lina; TAHRAOUI, Ahmed; KHEDDAM, Radja. Post-Flood Disaster Mapping Using Deep Neural Network. In: 2024 IEEE Mediterranean and Middle-East Geoscience and Remote Sensing Symposium (M2GARSS). [S. l.: s. n.], 2024. Pp. 182–186. DOI: 10.1109/M2GARSS57310.2024.10537322.

9 BRADSKI, G. The OpenCV Library. **Dr. Dobb's Journal of Software Tools**, 2000.

10 BREIMAN, Leo. Random Forests. **Machine Learning**, v. 45, n. 1, pp. 5–32, 2001. ISSN 0885-6125. DOI: 10.1023/A:1010933404324. Disponível em: <http://dx.doi.org/10.1023/A%3A1010933404324>.

11 CANNY, John. A Computational Approach to Edge Detection. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, PAMI-8, n. 6, pp. 679–698, 1986. DOI: 10.1109/TPAMI.1986.4767851.

12 CAO, Jingmiao et al. A Novel Wetland Classification Method Combined CNN and SVM Using Multi-Source Remote Sensing Images. In: IGARSS 2024 - 2024 IEEE International Geoscience and Remote Sensing Symposium. [S. l.: s. n.], 2024. Pp. 4793–4796. DOI: 10.1109/IGARSS53475.2024.10640705.

13 CEPED/UFSC. **Atlas Digital de Desastres no Brasil**. [S. l.: s. n.], 2025. Disponível em: <https://atlasdigital.mdr.gov.br/paginas/graficos.xhtml>.

14 _____. **Identificação de áreas susceptíveis a inundações no Estado do Rio de Janeiro com uso do método AHP e Sistemas de Informações Geográficas**. [S. l.: s. n.], 2025. Disponível em: <https://www.defesacivil.rj.gov.br/images/CEPEDEC/RELATORIOS/Relatrio---Mapa-de-Susceptibilidade-a-Inundaes-no-ERJ---MAR_2019.pdf>.

15 CHANDRAN, Divya V et al. Deep Learning-Based Flood Detection System Using Semantic Segmentation. In: 2024 7th International Conference on Circuit Power and Computing Technologies (ICCPCT). [S. l.: s. n.], 2024. v. 1, pp. 1584–1592. DOI: 10.1109/ICCPCT61902.2024.10673148.

16 CHEN, Liang-Chieh et al. **DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs**. [S. l.: s. n.], 2017. arXiv: 1606.00915 [cs.CV]. Disponível em: <https://arxiv.org/abs/1606.00915>.

17 CHEN, Tianqi; GUESTRIN, Carlos. XGBoost: A Scalable Tree Boosting System. In: PROCEEDINGS of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. San Francisco, California, USA: Association for Computing Machinery, 2016. (KDD '16), pp. 785–794. ISBN 9781450342322. DOI: 10.1145/2939672.2939785. Disponível em: <https://doi.org/10.1145/2939672.2939785>.

18 CHOLLET, Francois. **Deep Learning with Python**. [S. l.]: Manning Publications Co, 2019. ISBN 9781617294433.

19 CLARK, Alex. **Pillow (PIL Fork) Documentation**. [S. l.]: readthedocs, 2015. Disponível em: <https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf>.

20 COPERNICUS. **Introducing Sentinel-1**. [S. l.: s. n.], 2025. Acessado em: 18 de Janeiro, 2025. Disponível em: <https://www.esa.int/Applications/Observing_the_Earth/Copernicus/Sentinel-1/Introducing_Sentinel-1>.

21 _____. **S2 Mission**. [S. l.: s. n.], 2025. Acessado em: 18 de Janeiro, 2025. Disponível em: <https://sentiwiki.copernicus.eu/web/s2-mission>.

22 COVER, T.; HART, P. Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, v. 13, n. 1, pp. 21–27, 1967. DOI: 10.1109/TIT.1967.1053964.

23 CRAMER, J.S. **The Origins of Logistic Regression**. [S. l.], dez. 2002. Disponível em: <https://ideas.repec.org/p/tin/wpaper/20020119.html>.

24 DENG, Jia et al. Imagenet: A large-scale hierarchical image database. In: IEEE. 2009 IEEE conference on computer vision and pattern recognition. [S. l.: s. n.], 2009. Pp. 248–255.

25 DOERRY, Armin. Introduction to Synthetic Aperture Radar. In: 2019 IEEE Radar Conference (RadarConf). [S. l.: s. n.], 2019. Pp. 1–90. DOI: 10.1109/RADAR.2019.8835560.

26 DOGO, Eustace; AFOLABI, Oluwatobi; TWALA, Bhekisipho. On the Relative Impact of Optimizers on Convolutional Neural Networks with Varying Depth and Width for Image Classification. **Applied Sciences**, v. 12, p. 11976, nov. 2022. DOI: 10.3390/app122311976.

27 DOSOVITSKIY, Alexey et al. **An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale**. [S. l.: s. n.], 2021. arXiv: 2010.11929 [cs.CV].

28 DOZAT, Timothy. Incorporating Nesterov Momentum into Adam. In: PROCEEDINGS of the 4th International Conference on Learning Representations. [S. l.: s. n.], 2016. Pp. 1–4.

29 EVGENIOU, Theodoros; PONTIL, Massimiliano. Support Vector Machines: Theory and Applications. In: v. 2049, pp. 249–257. ISBN 978-3-540-42490-1. DOI: 10.1007/3-540-44673-7_12.

30 FAWCETT, Tom. An introduction to ROC analysis. **Pattern Recognition Letters**, v. 27, n. 8, pp. 861–874, 2006. ROC Analysis in Pattern Recognition. ISSN 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2005.10.010. Disponível em: <https://www.sciencedirect.com/science/article/pii/S016786550500303X>.

31 GESMUNDO, Andrea. **A Continual Development Methodology for Large-scale Multitask Dynamic ML Systems**. [S. l.: s. n.], 2022. arXiv: 2209.07326 [cs.LG].

32 GONÇALVES, André; RESENDE, Luis; CONCI, Aura. Comparing Deep Learning Models for Image Classification in Urban Flooding. In: 2024 31st International Conference on Systems, Signals and Image Processing (IWSSIP). [S. l.: s. n.], 2024. Pp. 1–5. DOI: 10.1109/IWSSIP62407.2024.10634034.

33 GONZALEZ, Rafael C.; WOODS, Richard E. **Digital Image Processing**. 4. ed. [S. l.]: Pearson, 2018.

34 GOODFELLOW, Ian; BENGIO, Yoshua; COURVILLE, Aaron. **Deep Learning**. [S. l.]: MIT Press, 2016. http://www.deeplearningbook.org.

35 HE, Kaiming et al. Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV). [S. l.: s. n.], 2017. Pp. 2980–2988. DOI: 10.1109/ICCV.2017.322.

36 HIDAYAT, Muh. Taufik et al. Enhanced Flood Detection on Highways: A Comparative Study of MobileNet and VGG16 CNN Models Based on CCTV Images. In: 2024 4th International Conference of Science and Information Technology in Smart Administration (ICSINTESA). [S. l.: s. n.], 2024. Pp. 125–130. DOI: 10.1109/ICSINTESA62455.2024.10747897.

37 HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long Short-Term Memory. **Neural Computation**, v. 9, n. 8, pp. 1735–1780, nov. 1997. ISSN 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. Disponível em: <https://doi.org/10.1162/neco.1997.9.8.1735>.

38 HOWARD, Andrew et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, abr. 2017. DOI: 10.48550/arXiv.1704.04861.

39 HUANG, Binbin et al. WaterDetectionNet: A New Deep Learning Method for Flood Mapping With SAR Image Convolutional Neural Network. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 17, pp. 14471–14485, 2024. DOI: 10.1109/JSTARS.2024.3440995.

40 HUANG, Gao et al. Densely connected convolutional networks. In: PROCEEDINGS of the IEEE Conference on Computer Vision and Pattern Recognition. [S. l.: s. n.], 2017.

41 HUBEL, D. H.; WIESEL, T. N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. **The Journal of Physiology**, v. 160, n. 1, pp. 106–154, 1962. DOI: https://doi.org/10.1113/jphysiol.1962.sp006837. eprint: https://physoc.onlinelibrary.wiley.com/doi/pdf/10.1113/jphysiol.1962.sp006837. Disponível em: <https://physoc.onlinelibrary.wiley.com/doi/abs/10.1113/jphysiol.1962.sp006837>.

42 IKOTUN, Abiodun M. et al. K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. **Information Sciences**, v. 622, pp. 178–210, 2023. ISSN 0020-0255. DOI: https://doi.org/10.1016/j.ins.2022.11.139. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0020025522014633>.

43 ISLAM, Md Azharul et al. An integrated convolutional neural network and sorting algorithm for image classification for efficient flood disaster management. **Decision Analytics Journal**, v. 7, p. 100225, 2023. ISSN 2772-6622. DOI: https://doi.org/10.1016/j.dajour.2023.100225. Disponível em: <https://www.sciencedirect.com/science/article/pii/S2772662223000656>.

44 JAMUNADEVI, .C et al. Application of Deep Learning Algorithm for Prediction of Flood Severity. In: pp. 1637–1642. DOI: 10.1109/ICSCSS60660.2024.10625025.

45 JONY, Rabiul Islam; WOODLEY, Alan; PERRIN, Dimitri. Flood Detection in Social Media Images using Visual Features and Metadata. In: 2019 Digital Image Computing: Techniques and Applications (DICTA). [S. l.: s. n.], 2019. Pp. 1–8. DOI: 10.1109/DICTA47822.2019.8946007.

46 KABIR, H M Dipu. **Reduction of Class Activation Uncertainty with Background Information**. [S. l.: s. n.], 2023. arXiv: 2305.03238 [cs.CV].

47 KINGMA, Diederik; BA, Jimmy. Adam: A Method for Stochastic Optimization. **International Conference on Learning Representations**, dez. 2014.

48 KOLESNIKOV, Alexander et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In.

49 KOLESNIKOV, Alexander et al. **Big Transfer (BiT): General Visual Representation Learning**. [S. l.: s. n.], 2020. arXiv: 1912.11370 [cs.CV].

50 KOOL, Juliette et al. Seasonal inundation dynamics and water balance of the Mara Wetland, Tanzania based on multi-temporal Sentinel-2 image classification. **International Journal of Applied Earth Observation and Geoinformation**, v. 109, p. 102766, 2022. ISSN 1569-8432. DOI: https://doi.org/10.1016/j.jag.2022.102766. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0303243422000927>.

51 KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet Classification with Deep Convolutional Neural Networks. In_____. **Advances in Neural Information Processing Systems**. [S. l.]: Curran Associates, Inc., 2012. v. 25. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf>.

52 LIM, Jae S. **Two-Dimensional Signal and Image Processing**. [S. l.]: Prentice-Hall, 1990.

53 LIU, Haiyang et al. A Comparison of Different Water Indices and Band Downscaling Methods for Water Bodies Mapping from Sentinel-2 Imagery at 10-M Resolution. **Water**, v. 14, n. 17, 2022. ISSN 2073-4441. DOI: 10.3390/w14172696. Disponível em: <https://www.mdpi.com/2073-4441/14/17/2696>.

54 MAO, Anqi; MOHRI, Mehryar; ZHONG, Yutao. Cross-entropy loss functions: theoretical analysis and applications. In: PROCEEDINGS of the 40th International Conference on Machine Learning. Honolulu, Hawaii, USA: JMLR.org, 2023. (ICML'23).

55 MICHELIN. **What is Aquaplaning and how to avoid it?** [S. l.: s. n.], 2024. Acessado em: 02 de Novembro, 2024. Disponível em: <https://www.michelin.co.uk/auto/advice/driving-tips/aquaplaning>.

56 MULTIMEDIA EVALUATION, MediaEval Benchmarking Initiative for. **The 2017 Multimedia Satellite Task: Emergency Response for Flooding Events**. [S. l.: s. n.], 2017. Acessado em: 02 de Novembro, 2024. Disponível em: <http://www.multimediaeval.org/mediaeval2017/multimediasatellite/>.

57 MURPHY, Kevin P. **Machine Learning: A Probabilistic Perspective**. 1. ed. [S. l.]: The MIT Press, 2012. (Adaptive Computation and Machine Learning).

58 MURTAGH, Fionn. Multilayer perceptrons for classification and regression. **Neurocomputing**, v. 2, n. 5, pp. 183–197, 1991. ISSN 0925-2312. DOI: https://doi.org/10.1016/0925-2312(91)90023-5. Disponível em: <https://www.sciencedirect.com/science/article/pii/0925231291900235>.

59 MYIN, Moe Moe; THEIN, Thin Lai Lai. Flood Mapping System of Flooding Prone Area in Myanmar. In: 2024 IEEE Conference on Computer Applications (ICCA). [S. l.: s. n.], 2024. Pp. 1–5. DOI: 10.1109/ICCA62361.2024.10532869.

60 OHTA, Noboru; ROBERTSON, Alan. **Colorimetry**: Fundamentals and Applications. 1. ed. [S. l.]: Joh Wiley & Sons, 2005.

61 PALLY, R.J.; SAMADI, S. Application of image processing and convolutional neural networks for flood image classification and semantic segmentation. **Environmental Modelling and Software**, v. 148, p. 105285, 2022. ISSN 1364-8152. DOI: https://doi.org/10.1016/j.envsoft.2021.105285. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1364815221003273>.

62 PECH-MAY, Fernando et al. Segmentation and Visualization of Flooded Areas Through Sentinel-1 Images and U-Net. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 17, pp. 8996–9008, 2024. DOI: 10.1109/JSTARS.2024.3387452.

63 PIEDAD, Eduardo Jr et al. Intelligent Flood Detection using Traffic Surveillance Images based on Convolutional Neural Network and Image Parsing. In: 2022 IEEE International Conference on Computing (ICOCO). [S. l.: s. n.], 2022. Pp. 220–225. DOI: 10.1109/ICOCO56118.2022.10031718.

64 RAO, Chengping; LIU, Yang. Three-dimensional convolutional neural network (3D-CNN) for heterogeneous material homogenization. **Computational Materials Science**, v. 184, p. 109850, 2020. ISSN 0927-0256. DOI: https://doi.org/10.1016/j.commatsci.2020.109850. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0927025620303414>.

65 REDDY, K.Venkata Vinay Kumar; VIMAL, V.R. A Novel Approach on Improved Segmentation and Classification of Remote Sensing Images using AlexNet Compared over Linear Discriminant Analysis with Improved Accuracy. In: 2024 Second International Conference on Advances in Information Technology (ICAIT). [S. l.: s. n.], 2024. v. 1, pp. 1–6. DOI: 10.1109/ICAIT61638.2024.10690378.

66 REZATOFIGHI, Hamid et al. Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, jun. 2019. Pp. 658–666. DOI: 10.1109/CVPR.2019.00075. Disponível em: <https://doi.ieeecomputersociety.org/10.1109/CVPR.2019.00075>.

67 RIO, Prefeitura do. **PLANO DIRETOR DE MANEJO DE ÁGUAS PLUVIAIS DA CIDADE DO RIO DE JANEIRO**. [S. l.: s. n.], 2013. Disponível em: <http://www.rio.rj.gov.br/dlstatic/10112/8940582/4249724/RA0027.RA.3775_RELATORIOSINTESEPDMAP.pdf>.

68 RONNEBERGER, Olaf; FISCHER, Philipp; BROX, Thomas. U-Net: Convolutional Networks for Biomedical Image Segmentation. In_____. **Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015**. Cham: Springer International Publishing, 2015. Pp. 234–241. ISBN 978-3-319-24574-4.

69 RUDER, Sebastian. An overview of gradient descent optimization algorithms, set. 2016. DOI: 10.48550/arXiv.1609.04747.

70 RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning representations by back-propagating errors. **Nature**, v. 323, pp. 533–536, 1986. Disponível em: <https://api.semanticscholar.org/CorpusID:205001834>.

71 SANDLER, Mark et al. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. [S. l.: s. n.], 2018. Pp. 4510–4520. DOI: 10.1109/CVPR.2018.00474.

72 SAZARA, Cem; CETIN, Mecit; IFTEKHARUDDIN, Khan M. Detecting floodwater on roadways from image data with handcrafted features and deep transfer learning. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). [S. l.: s. n.], 2019. Pp. 804–809. DOI: 10.1109/ITSC.2019.8917368.

73 SCHMIDHUBER, Jürgen. Deep learning in neural networks: An overview. **Neural Networks**, v. 61, pp. 85–117, 2015. ISSN 0893-6080. DOI: https://doi.org/10.1016/j.neunet.2014.09.003. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0893608014002135>.

74 SGHAIER, Souhir et al. Natural disasters detection and classification based on deep learning. In: 2023 3rd International Conference on Computing and Information Technology (ICCIT). [S. l.: s. n.], 2023. Pp. 334–339. DOI: 10.1109/ICCIT58132.2023.10273976.

75 SHAH, Aditya et al. Machine Learning for Flood Susceptibility Mapping of the Chennai Floods of 2023. In: pp. 1–6. DOI: 10.1109/SPICES62143.2024.10779868.

76 SHAPIRO, Linda; STOCKMAN, George. **Computer Vision**. 1. ed. [S. l.]: Pearson, 2001.

77 SIDDIQUE, Mohammed; AHMED, Tasneem; HUSAIN, Mohd Shahid. An Integrated Image Classification Approach to Detect the Flood Prone Areas using Sentinel-1 Images. In: 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom). [S. l.: s. n.], 2023. Pp. 655–660.

78 SIMONYAN, K; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. In: pp. 1–14.

79 SUDIANA, Dodi et al. Performance Evaluation of 3-D Convolutional Neural Network for Multitemporal Flood Classification Framework With Synthetic Aperture Radar Image Data. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing**, v. 18, pp. 3198–3207, 2025. DOI: 10.1109/JSTARS.2024.3519523.

80 SZEGEDY, Christian et al. Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, jun. 2015. Pp. 1–9. DOI: 10.1109/CVPR.2015.7298594. Disponível em: <https://doi.ieeecomputersociety.org/10.1109/CVPR.2015.7298594>.

81 SZEGEDY, Christian et al. Rethinking the Inception Architecture for Computer Vision. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Los Alamitos, CA, USA: IEEE Computer Society, jun. 2016. Pp. 2818–2826. DOI: 10.1109/CVPR.2016.308. Disponível em: <https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.308>.

82 THARWAT, Alaa et al. Linear discriminant analysis: A detailed tutorial. **Ai Communications**, v. 30, pp. 169–190, mai. 2017. DOI: 10.3233/AIC-170729.

83 TOUFIQUE, S.M et al. Implementing Machine Learning Techniques to Forecast Floods in Bangladesh. In: pp. 1–6. DOI: 10.1109/ICECET61485.2024.10698703.

84 UNDRR. **GAR 2025 Hazards: Floods**. [S. l.: s. n.], 2025. Disponível em: <https://www.undrr.org/gar/gar2025/hazard-exploration/floods>.

85 _____. **GAR 2025: Map explorations**. [S. l.: s. n.], 2025. Disponível em: <https://www.undrr.org/gar/gar2025/map-exploration#gdpClimate>.

86 VAN ROSSUM, Guido; DRAKE JR, Fred L. **Python tutorial**. [S. l.]: Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands, 1995.

87 VASWANI, Ashish et al. Attention is All you Need. In_____. **Advances in Neural Information Processing Systems**. [S. l.]: Curran Associates, Inc., 2017. v. 30. Disponível em: <https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>.

88 VIMALA, .M et al. Flood Prediction using Supervised Machine Learning Algorithms. In: pp. 1093–1096. DOI: 10.1109/ICOSEC61587.2024.10722348.

89 VINEETH, V; NEEBA, E A. Flood Detection using Deep Learning. In: 2021 International Conference on Advances in Computing and Communications (ICACC). [S. l.: s. n.], 2021. Pp. 1–5. DOI: 10.1109/ICACC-202152719.2021.9708240.

90 XU, Hanqiu. Modification of normalised difference water index (NDWI) to enhance open water features in remotely sensed imagery. **International Journal of Remote Sensing**, Taylor & Francis, v. 27, n. 14, pp. 3025–3033, 2006. DOI: 10.1080/01431160600589179. eprint: https://doi.org/10.1080/01431160600589179. Disponível em: <https://doi.org/10.1080/01431160600589179>.

# APÊNDICE A – Tabela de quantidade de images de cada classe para cada câmera identificada pelo seu código

Tabela 10: Quantidade de images de cada classe para cada câmera identificada pelo seu código

| **Câmera** | **Classe 0** | **Classe 1** | **Câmera** | **Classe 0** | **Classe 1** |
|---|---|---|---|---|---|
| 1 | 936 | 0 | 1294 | 45 | 0 |
| 5 | 5 | 0 | 1303 | 87 | 0 |
| 13 | 2572 | 0 | 1311 | 0 | 90 |
| 14 | 180 | 15 | 1314 | 9 | 0 |
| 16 | 11 | 0 | 1338 | 15 | 0 |
| 17 | 564 | 0 | 1383 | 900 | 0 |
| 18 | 3098 | 0 | 1385 | 90 | 0 |
| 19 | 900 | 0 | 1387 | 180 | 0 |
| 20 | 235 | 0 | 1389 | 6 | 0 |
| 22 | 495 | 0 | 1390 | 0 | 16 |
| 24 | 4 | 0 | 1391 | 116 | 0 |
| 25 | 9 | 0 | 1392 | 343 | 0 |
| 26 | 24 | 0 | 1393 | 690 | 858 |
| 27 | 18 | 0 | 1397 | 90 | 0 |
| 31 | 18 | 0 | 1403 | 277 | 504 |
| 32 | 963 | 495 | 1404 | 282 | 1622 |
| 35 | 45 | 0 | 1405 | 225 | 1397 |
| 36 | 45 | 0 | 1410 | 18 | 54 |
| 37 | 15 | 0 | 1426 | 0 | 9 |
| 38 | 2835 | 945 | 1427 | 15 | 0 |

| **Câmera** | **Classe 0** | **Classe 1** | **Câmera** | **Classe 0** | **Classe 1** |
|---|---|---|---|---|---|
| 44 | 475 | 648 | 1428 | 0 | 45 |
| 45 | 522 | 0 | 1430 | 232 | 180 |
| 46 | 11 | 0 | 1431 | 277 | 73 |
| 48 | 45 | 0 | 1459 | 1851 | 0 |
| 49 | 0 | 405 | 1460 | 2756 | 0 |
| 58 | 2350 | 0 | 1461 | 5383 | 0 |
| 61 | 49 | 0 | 1462 | 5604 | 0 |
| 65 | 4446 | 0 | 1468 | 724 | 0 |
| 68 | 192 | 0 | 1469 | 665 | 0 |
| 69 | 45 | 0 | 1474 | 1048 | 3448 |
| 74 | 135 | 0 | 1475 | 884 | 3737 |
| 77 | 366 | 290 | 1476 | 394 | 0 |
| 79 | 45 | 0 | 1477 | 495 | 0 |
| 88 | 96 | 0 | 1487 | 1836 | 1742 |
| 90 | 540 | 0 | 1492 | 45 | 0 |
| 92 | 4076 | 276 | 1506 | 720 | 0 |
| 93 | 10 | 0 | 1507 | 450 | 0 |
| 94 | 225 | 225 | 1509 | 0 | 180 |
| 96 | 1504 | 0 | 1514 | 360 | 163 |
| 97 | 1628 | 0 | 1515 | 0 | 135 |
| 98 | 11 | 0 | 1524 | 387 | 1122 |
| 99 | 15 | 0 | 1525 | 382 | 1375 |
| 103 | 0 | 15 | 1528 | 2361 | 0 |
| 105 | 50 | 0 | 1529 | 1103 | 0 |
| 106 | 212 | 0 | 1531 | 135 | 0 |
| 107 | 142 | 0 | 1534 | 28 | 3234 |
| 112 | 420 | 0 | 1535 | 135 | 242 |
| 114 | 924 | 225 | 1538 | 2028 | 5640 |
| 115 | 900 | 0 | 1539 | 2266 | 45 |
| 116 | 9 | 0 | 1544 | 0 | 45 |
| 117 | 180 | 0 | 1545 | 9 | 0 |
| 120 | 443 | 0 | 1546 | 1035 | 0 |
| 122 | 45 | 0 | 1547 | 482 | 0 |
| 124 | 14 | 0 | 1548 | 15 | 0 |

| **Câmera** | **Classe 0** | **Classe 1** | **Câmera** | **Classe 0** | **Classe 1** |
|---|---|---|---|---|---|
| 126 | 18 | 0 | 1551 | 9 | 0 |
| 127 | 1459 | 0 | 1556 | 826 | 0 |
| 131 | 45 | 0 | 1557 | 867 | 0 |
| 136 | 135 | 585 | 1566 | 180 | 0 |
| 138 | 630 | 0 | 1574 | 5 | 0 |
| 142 | 39 | 0 | 1575 | 1592 | 0 |
| 145 | 135 | 0 | 1576 | 1102 | 0 |
| 147 | 45 | 0 | 1577 | 9 | 0 |
| 149 | 45 | 0 | 1583 | 945 | 0 |
| 150 | 50 | 0 | 1584 | 41 | 0 |
| 151 | 277 | 0 | 1585 | 298 | 0 |
| 153 | 45 | 0 | 1587 | 495 | 0 |
| 158 | 180 | 135 | 1588 | 1512 | 0 |
| 159 | 0 | 90 | 1591 | 495 | 0 |
| 160 | 495 | 540 | 1598 | 1970 | 0 |
| 161 | 590 | 0 | 1599 | 52 | 0 |
| 169 | 418 | 0 | 1600 | 1963 | 0 |
| 173 | 45 | 0 | 1601 | 1956 | 0 |
| 174 | 457 | 0 | 1603 | 495 | 0 |
| 175 | 90 | 0 | 1606 | 127 | 109 |
| 176 | 0 | 135 | 1612 | 20 | 19 |
| 178 | 6 | 0 | 1613 | 16 | 0 |
| 179 | 45 | 0 | 1623 | 8 | 0 |
| 182 | 312 | 0 | 1629 | 9 | 0 |
| 183 | 495 | 0 | 1635 | 1076 | 924 |
| 190 | 45 | 0 | 1636 | 977 | 987 |
| 192 | 90 | 0 | 1637 | 37 | 0 |
| 196 | 45 | 0 | 1638 | 4455 | 0 |
| 201 | 450 | 0 | 1639 | 5158 | 0 |
| 203 | 1160 | 0 | 1644 | 630 | 0 |
| 206 | 0 | 2 | 1646 | 0 | 1063 |
| 207 | 15 | 0 | 1648 | 365 | 1148 |
| 209 | 45 | 0 | 1649 | 448 | 1279 |
| 217 | 810 | 0 | 1651 | 0 | 720 |

| **Câmera** | **Classe 0** | **Classe 1** | **Câmera** | **Classe 0** | **Classe 1** |
|---|---|---|---|---|---|
| 218 | 45 | 0 | 1661 | 495 | 0 |
| 226 | 3414 | 0 | 1671 | 2700 | 630 |
| 230 | 225 | 0 | 1676 | 6 | 0 |
| 235 | 7889 | 3021 | 1677 | 6 | 0 |
| 240 | 611 | 0 | 1690 | 15 | 0 |
| 243 | 45 | 0 | 1701 | 9 | 0 |
| 247 | 7 | 0 | 1703 | 18 | 0 |
| 249 | 3460 | 0 | 1726 | 2925 | 0 |
| 255 | 675 | 0 | 1727 | 45 | 45 |
| 257 | 45 | 0 | 1745 | 24 | 0 |
| 259 | 1890 | 0 | 1821 | 10 | 0 |
| 261 | 31 | 529 | 1857 | 9 | 0 |
| 262 | 434 | 624 | 1881 | 2560 | 540 |
| 263 | 60 | 0 | 1882 | 0 | 45 |
| 267 | 6321 | 288 | 1884 | 45 | 45 |
| 268 | 69 | 19 | 1887 | 7 | 0 |
| 269 | 1107 | 817 | 1907 | 0 | 45 |
| 273 | 3632 | 0 | 1922 | 0 | 9 |
| 278 | 14305 | 10310 | 1926 | 7 | 0 |
| 286 | 9 | 0 | 1954 | 42 | 0 |
| 290 | 25 | 0 | 1965 | 765 | 9 |
| 294 | 45 | 0 | 1972 | 90 | 0 |
| 298 | 2207 | 6045 | 1981 | 45 | 0 |
| 299 | 5402 | 0 | 1985 | 9 | 0 |
| 300 | 9 | 0 | 1994 | 1708 | 122 |
| 301 | 0 | 78 | 2001 | 45 | 0 |
| 302 | 26 | 451 | 2002 | 26 | 270 |
| 303 | 135 | 0 | 2004 | 9 | 0 |
| 310 | 528 | 135 | 2005 | 431 | 0 |
| 312 | 0 | 595 | 2015 | 772 | 0 |
| 313 | 850 | 1558 | 2017 | 360 | 97 |
| 315 | 45 | 0 | 2038 | 15 | 0 |
| 326 | 10354 | 1112 | 2040 | 60 | 0 |
| 328 | 45 | 0 | 2047 | 89 | 0 |

| **Câmera** | **Classe 0** | **Classe 1** | **Câmera** | **Classe 0** | **Classe 1** |
|---|---|---|---|---|---|
| 331 | 495 | 0 | 2051 | 45 | 0 |
| 333 | 15 | 0 | 2052 | 374 | 0 |
| 339 | 331 | 125 | 2053 | 374 | 0 |
| 341 | 1245 | 135 | 2054 | 412 | 0 |
| 342 | 109 | 0 | 2055 | 450 | 0 |
| 345 | 0 | 45 | 2076 | 6 | 0 |
| 357 | 4 | 0 | 2088 | 14 | 0 |
| 358 | 38 | 0 | 2105 | 5 | 0 |
| 361 | 45 | 0 | 2107 | 54 | 0 |
| 363 | 1440 | 0 | 2126 | 8 | 0 |
| 365 | 951 | 130 | 2138 | 0 | 45 |
| 367 | 0 | 101 | 2140 | 20 | 0 |
| 370 | 430 | 0 | 2156 | 315 | 0 |
| 378 | 0 | 90 | 2157 | 315 | 0 |
| 384 | 162 | 225 | 2158 | 277 | 0 |
| 390 | 225 | 0 | 2160 | 0 | 9 |
| 392 | 0 | 24 | 2161 | 9 | 0 |
| 398 | 0 | 225 | 2165 | 1506 | 0 |
| 404 | 405 | 0 | 2166 | 1463 | 0 |
| 407 | 52 | 0 | 2167 | 1439 | 0 |
| 410 | 45 | 0 | 2190 | 30 | 0 |
| 414 | 225 | 0 | 2194 | 45 | 0 |
| 416 | 45 | 0 | 2195 | 0 | 9 |
| 418 | 232 | 0 | 2202 | 0 | 7 |
| 421 | 233 | 0 | 2204 | 450 | 0 |
| 423 | 0 | 180 | 2205 | 434 | 0 |
| 425 | 90 | 0 | 2206 | 922 | 0 |
| 430 | 1155 | 585 | 2214 | 0 | 180 |
| 432 | 467 | 0 | 2216 | 540 | 45 |
| 442 | 202 | 630 | 2218 | 45 | 0 |
| 446 | 21 | 0 | 2219 | 4031 | 0 |
| 448 | 7 | 0 | 2221 | 180 | 225 |
| 454 | 322 | 0 | 2228 | 9 | 0 |
| 455 | 367 | 0 | 2233 | 0 | 9 |

| **Câmera** | **Classe 0** | **Classe 1** | **Câmera** | **Classe 0** | **Classe 1** |
|---|---|---|---|---|---|
| 457 | 0 | 14 | 2238 | 405 | 0 |
| 460 | 344 | 0 | 2239 | 0 | 180 |
| 472 | 14 | 0 | 2243 | 0 | 275 |
| 480 | 45 | 0 | 2246 | 758 | 0 |
| 482 | 3097 | 0 | 2268 | 101 | 0 |
| 484 | 135 | 0 | 2270 | 90 | 225 |
| 500 | 0 | 5 | 2277 | 18 | 630 |
| 516 | 0 | 7 | 2278 | 90 | 45 |
| 517 | 6 | 0 | 2280 | 630 | 225 |
| 532 | 90 | 0 | 2296 | 250 | 0 |
| 547 | 135 | 0 | 2301 | 90 | 0 |
| 550 | 945 | 0 | 2308 | 9 | 0 |
| 561 | 45 | 810 | 2327 | 45 | 0 |
| 571 | 0 | 7 | 2347 | 720 | 0 |
| 572 | 0 | 108 | 2349 | 0 | 360 |
| 596 | 45 | 45 | 2360 | 45 | 0 |
| 600 | 10 | 0 | 2361 | 45 | 45 |
| 661 | 131 | 0 | 2365 | 30 | 374 |
| 662 | 147 | 0 | 2368 | 0 | 156 |
| 663 | 61 | 0 | 2379 | 42 | 0 |
| 664 | 136 | 0 | 2387 | 45 | 0 |
| 680 | 315 | 0 | 2396 | 45 | 225 |
| 681 | 19 | 0 | 2401 | 39 | 0 |
| 705 | 180 | 0 | 2426 | 135 | 0 |
| 724 | 270 | 45 | 2431 | 0 | 4 |
| 758 | 269 | 0 | 2445 | 15 | 0 |
| 766 | 810 | 10 | 2460 | 9 | 0 |
| 767 | 810 | 0 | 3016 | 14 | 0 |
| 770 | 1796 | 0 | 3019 | 6 | 0 |
| 782 | 45 | 0 | 3100 | 0 | 45 |
| 793 | 45 | 0 | 3103 | 0 | 218 |
| 842 | 1706 | 0 | 3104 | 90 | 0 |
| 846 | 945 | 0 | 3114 | 1063 | 0 |
| 869 | 900 | 0 | 3127 | 1181 | 0 |

| **Câmera** | **Classe 0** | **Classe 1** | **Câmera** | **Classe 0** | **Classe 1** |
|---|---|---|---|---|---|
| 894 | 45 | 0 | 3133 | 720 | 90 |
| 923 | 60 | 0 | 3134 | 45 | 450 |
| 963 | 495 | 0 | 3135 | 2706 | 135 |
| 1000 | 18 | 0 | 3147 | 45 | 0 |
| 1008 | 74 | 0 | 3162 | 15 | 0 |
| 1010 | 270 | 0 | 3181 | 7378 | 0 |
| 1014 | 11 | 0 | 3197 | 45 | 0 |
| 1015 | 100 | 0 | 3206 | 0 | 6 |
| 1018 | 0 | 4 | 3215 | 24 | 0 |
| 1020 | 64 | 0 | 3217 | 9 | 0 |
| 1026 | 838 | 0 | 3220 | 141 | 45 |
| 1027 | 928 | 0 | 3221 | 0 | 45 |
| 1061 | 135 | 0 | 3224 | 450 | 0 |
| 1073 | 1269 | 0 | 3254 | 0 | 15 |
| 1080 | 810 | 0 | 3255 | 0 | 24 |
| 1083 | 1495 | 1134 | 3260 | 405 | 0 |
| 1092 | 322 | 0 | 3261 | 8720 | 0 |
| 1099 | 0 | 720 | 3262 | 426 | 0 |
| 1110 | 9 | 0 | 3263 | 957 | 0 |
| 1111 | 54 | 0 | 3264 | 93 | 0 |
| 1119 | 264 | 1203 | 3265 | 2304 | 0 |
| 1120 | 169 | 25 | 3266 | 45 | 0 |
| 1121 | 171 | 0 | 3268 | 140 | 0 |
| 1123 | 945 | 0 | 3269 | 45 | 0 |
| 1128 | 0 | 14 | 3270 | 1042 | 90 |
| 1132 | 0 | 90 | 3278 | 35 | 0 |
| 1138 | 32 | 1052 | 3286 | 60 | 0 |
| 1139 | 17 | 1175 | 3287 | 0 | 15 |
| 1141 | 945 | 0 | 3290 | 1191 | 306 |
| 1144 | 45 | 0 | 3318 | 3060 | 0 |
| 1147 | 195 | 315 | 3323 | 90 | 495 |
| 1156 | 207 | 0 | 3324 | 45 | 0 |
| 1157 | 189 | 0 | 3346 | 15 | 0 |
| 1161 | 208 | 0 | 3347 | 45 | 270 |

| **Câmera** | **Classe 0** | **Classe 1** | **Câmera** | **Classe 0** | **Classe 1** |
|---|---|---|---|---|---|
| 1162 | 203 | 0 | 3358 | 180 | 0 |
| 1169 | 17 | 1621 | 3448 | 45 | 0 |
| 1170 | 18 | 1468 | 3533 | 315 | 0 |
| 1182 | 45 | 225 | 3545 | 277 | 0 |
| 1215 | 0 | 45 | 3550 | 0 | 15 |
| 1216 | 54 | 90 | 3558 | 270 | 0 |
| 1220 | 8 | 0 | 3581 | 90 | 0 |
| 1221 | 363 | 0 | 3594 | 0 | 6 |
| 1222 | 992 | 0 | 3596 | 9 | 0 |
| 1223 | 45 | 0 | 3600 | 0 | 14 |
| 1267 | 45 | 0 | 3627 | 0 | 135 |
| 1271 | 45 | 0 | 3638 | 45 | 0 |