Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

# Transfer learning e convolutional neural networks para a classificação de imagens e reconhecimento de objetos no âmbito da perícia criminal

Juliano Rodrigues de Almeida

Dissertação apresentada como requisito parcial para qualificação do
Mestrado Profissional em Computação Aplicada

Orientador
Prof. Dr. Alexandre Zaghetto

Brasília
2020.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

# Transfer learning e convolutional neural networks para a classificação de imagens e reconhecimento de objetos no âmbito da perícia criminal

Juliano Rodrigues de Almeida

Dissertação apresentada como requisito parcial para qualificação do
Mestrado Profissional em Computação Aplicada

Prof. Dr. Alexandre Zaghetto (Orientador)
CIC/UnB

Prof. Dr. Marcelo Ladeira
Universidade de Brasília

Prof. Dr. Remis Balaniuk
Universidade Católica de Brasília

Prof. Dr. Luís Paulo Faina Garcia
Universidade de Brasília

do Programa de Pós-graduação em Computação Aplicada

Brasília, 27 de agosto de 2020.

# Resumo

A popularização do uso de aparelhos eletrônicos, como *smartphones, tablets* e computadores, aliado ao aumento da capacidade de armazenamento desses dispositivos, está impactando a área da perícia criminal. Cada vez mais tempo e experiência são exigidos na análise de evidências digitais na investigação criminal. Diante desse cenário, técnicas de aprendizado de máquina estão sendo implantadas no campo da perícia criminal, a fim de auxiliar a obtenção de evidências relevantes de forma mais célere e uniforme. Este trabalho apresenta um estudo que utiliza *Transfer Learning* (TF) e *Convolutional Neural Networks* (CNN) na classificação de imagens e no reconhecimento de objetos de interesse na área da perícia criminal. O modelo para classificação de imagens criado a partir de CNN obteve, para os conjuntos de testes, acurácia de 98% na classificação de imagens contendo documentos e não documentos, e 99% de acurácia na classificação entre documentos de identificação e documentos gerais. No que tange à detecção de armas em arquivos de imagem, o modelo criado utilizando a arquitetura YOLOv3 obteve mAP de 0,85. Os classificadores desenvolvidos foram integrados ao *software* de perícia computacional Autopsy, fornecendo acurácia de 97,9% na classificação de imagens contendo documentos. Foi possível, ainda, integrar a detecção de pessoas, carros e armas a essa ferramenta, permitindo identificar corretamente armas em 77,2% das imagens testadas. Por meio dos resultados obtidos com a integração com o *software* Autopsy, verifica-se o potencial prático deste projeto para realizar classificações e detecções de objetos em imagens objeto de perícia criminal.

**Palavras-chave:** Transferência de aprendizado, Redes Neurais Convolutionais (RNC), classificação de imagens, reconhecimento de objetos, YOLOv3, processamento de imagens forenses.

# Abstract

The popularization of electronic equipments such as smartphones, tablets and computers combined with the increase in the individual capacity of these devices have impacted the digital forensic field. More time and expertise are required in the analysis of digital evidence in the criminal investigation. Given this scenario, machine learning techniques are being studied and deployed in the realm of criminal investigation aimed at producing the forensic evidence in a faster and more uniform way. This project presents a category-based image classification and object recognition study using Convolutional Neural Networks (CNN) and Transfer Learning (TF).The image classification model created from CNN obtained, for the test sets, 98% of accuracy in the classification of images containing documents and not documents, and 99% of accuracy in the classification between identification documents and general documents. Regarding the detection of weapons in image files, the model created using the YOLOv3 architecture obtained a mAP of 0.85. The developed classifiers were integrated with the Autopsy computer expertise software, providing 97.9% accuracy in the classification of images containing documents. It was also possible to integrate the detection of people, cars and weapons into this tool, allowing to correctly identify weapons in 77.2% of the tested images. Through the results obtained with the integration with the Autopsy software, it is possible to verify the practical potential of this project to carry out classifications and detections of objects in images subject to criminal investigations.

**Keywords:** Transfer learning, Convolutional Neural Networks (CNN), image classification, object recognition, YOLOv3, forensic image processing.

# Sumário

# Lista de Figuras

# Lista de Tabelas

# Capítulo 1

# Introdução

Este Capítulo apresenta a contextualização do tema deste trabalho, bem como a justificativa para um estudo na área de análise de imagens no campo da perícia criminal. São apresentados os objetivos geral e específicos, assim como a estrutura do presente trabalho.

## 1.1 Contextualização

A Constituição Federal do Brasil, de 1988, dispõe que a segurança pública é dever do Estado, direito e responsabilidade de todos, sendo aplicada para a preservação da ordem pública e da incolumidade das pessoas e do patrimônio. Essa atividade é exercida, dentre outros órgãos, pela polícia civil, cabendo-lhe a apuração de infrações penais [1].

De acordo com o Código de Processo Penal [2], "quando uma infração deixar vestígios, será indispensável o exame de corpo de delito, direto ou indireto, não podendo supri-lo a confissão do acusado". Esse exame e outras perícias devem ser realizados por perito oficial, o qual elaborará laudo pericial, em que descreverá minuciosamente o que examinar, bem como responderá aos quesitos formulados.

Assim, a atividade pericial é desempenhada por perito oficial, auxiliar da justiça com conhecimento especializado em determinada área do conhecimento que, por meio de suas experiências, sua perspicácia e seu espírito investigativo, auxiliará na comprovação da verdade dos fatos. A perícia criminal envolve a realização de exames em diversos campos como, por exemplo, computação, engenharia, química, contabilidade, balística, documentoscopia e grafoscopia.

A perícia computacional forense, também chamada de computação forense ou perícia digital, tem como objetivo principal determinar a dinâmica, a materialidade e a autoria de ilícitos ligados à área cibernética [3]. Os crimes cibernéticos podem ser classificados em duas categorias: crimes *cyber-enabled* e crimes *cyber-dependent*. O primeiro tipo caracteriza-se por utilizar tecnologias da informação e comunicação como ferramenta

acessória para o aumento de seu alcance, ao passo que a segunda categoria caracteriza-se pelos crimes que são viabilizados unicamente com o emprego de tecnologias de informação e comunicação [4]. Desse modo, a perícia computacional é realizada em aparelhos não apenas supostamente utilizados em crimes propriamente cibernéticos, como também em dispositivos eletrônicos que possam fornecer subsídios às investigações de quaisquer crimes.

Nos últimos anos, tem-se verificado um grande aumento na quantidade de *desktops*, *notebooks*, *tablets* e celulares comercializados no Brasil. A expectativa é que o número de *desktops*, *tablets* e *notebooks* chegue a 210 milhões de unidades entre os anos de 2020 e 2022 [5], sendo que em 2018 já havia cerca de 174 milhões de *smartphones*, representando mais 80% da população brasileira [5]. Apesar dos benefícios trazidos pela difusão desses dispositivos, essas tecnologias estão sendo cada vez mais exploradas para o cometimento de crimes. Estima-se que, em âmbito global, crimes cibernéticos poderão provocar prejuízos de cerca de US$ 6 trilhões em 2021 [6].

Nesse cenário, a Seção de Perícias em Informática (SPI) da Polícia Civil do Distrito Federal (PCDF) se depara com uma demanda crescente para perícias de *desktops*, *notebooks* e dispositivos móveis. São exigidos tempo, conhecimento especializado e ferramentas adequadas para a extração e análise de vestígios relacionados ao crime sob investigação, bem como a elaboração de laudos e relatórios determinados a substanciar a autoridade policial em suas tomadas de decisões, e o magistrado em seus julgamentos.

## 1.2 Definição do problema

Um problema recorrente no trabalho pericial da Seção de Perícias em Informática (SPI) da Polícia Civil do Distrito Federal (PCDF) está na necessidade de analisar manualmente grandes quantidades de arquivos, principalmente imagens. Os principais *softwares* forenses comerciais e institucionais são capazes de extrair arquivos de mídia de dispositivos, porém geralmente não aplicam nenhum tipo de categorização ou reconhecimento automático de elementos de interesse pericial possivelmente contidos nesses arquivos. Assim, a seleção de arquivos de imagem contendo pessoas, veículos, documentos digitalizados e armas, por exemplo, ou outros elementos de interesse, fica a cargo do perito criminal, o que pode demandar tempo considerável de sua atividade pericial.

Esse problema, que é o tempo excessivo para a conclusão de exames periciais devido à falta de ferramentas para categorização de arquivos de imagem de acordo com o seu conteúdo, e reconhecimento automático de elementos, é verificado em praticamente todos os exames os quais envolvam uma quantidade massiva de arquivos de imagem.

## 1.3 Justificativa do trabalho

Com a crescente utilização de *desktops, notebooks, tablets, pendrives* e *smartphones*, aliado ao incremento substancial em suas capacidades de armazenamento, as perícias nesses dispositivos estão demandando cada vez mais tempo e recursos. Antigos equipamentos com capacidade de armazenamento de 4 *gigabytes*, por exemplo, poderiam exigir poucos minutos de análise, enquanto que dispositivos com produção recente, comumente com 1 *terabyte* de armazenamento ou mais, podem requerer até dias de análise manual.

As atuais abordagens não são as mais eficientes para lidar com grandes quantidades de dados no contexto da computação forense. Essas abordagens baseiam-se, principalmente, na pesquisa por palavras chaves, correlações de *logs* de eventos, visualização de dados e exploração manual. Assim, há necessidade de adoção de modelos mais eficientes capazes de trabalhar com essa vasta quantidade de informações, a fim de processar e localizar evidências digitais [7].

Uma solução que categorize arquivos de imagem de acordo com o seu conteúdo (desenhos, pessoas, lugares, documentos digitalizados, etc.), bem como que seja capaz de reconhecer elementos de interesse (veículo, arma, possíveis drogas, etc.) seria capaz de dinamizar grande parte do trabalho pericial, fornecendo respostas mais rápidas às investigações criminais, facilitando e agilizando a elucidação de crimes como homicídios, latrocínios, tráfico de drogas e pedofilia, por exemplo. Tais resultados impactariam diretamente não apenas a seção onde esse modelo de atividade poderá ser utilizado, como também todos os órgãos requisitantes do trabalho técnico-pericial desenvolvido na SPI, especialmente as delegacias integrantes da PCDF, e Ministério Público.

O perito criminal que atua na análise de dispositivos computacionais poderá ter parte do seu trabalho dinamizado, por meio da categorização de imagens relevantes e de interesse pericial. Com uma produção técnico-pericial mais célere e eficiente, as investigações criminais poderão fornecer respostas mais rápidas a questões como autoria e dinâmica de eventos em atos delituosos. Essa maior agilidade no trabalho pericial impactará positivamente nas ações da própria instituição e dos órgãos solicitantes na resolução de crimes, contribuindo para o aumento da sensação de justiça.

## 1.4 Objetivos

Esta Seção mostra o objetivo geral do trabalho, bem como seus objetivos específicos. A partir do cumprimento dos objetivos específicos elencados, é formada a base necessária para o atingimento do objetivo geral proposto.

### 1.4.1 Objetivo geral

Desenvolver uma ferramenta para a classificação de imagens e detecção de objetos no âmbito da perícia criminal.

### 1.4.2 Objetivos específicos

O objetivo geral se desdobra nos seguintes objetivos específicos:

- Construir uma base de publicações científicas representativa do estado da arte nas áreas de classificação de imagens e detecção/reconhecimento de objetos.
- Compreender o processo de classificação de imagens e de reconhecimento de elementos em imagens.
- Construir uma base de imagens contendo os elementos de interesse do presente trabalho.
- Descobrir os modelos mais indicados para a solução do problema abordado.

## 1.5 Impacto esperado

Sob o ponto de vista *profissional*, espera-se que a partir deste estudo seja desenvolvida ou aperfeiçoada uma ferramenta que auxilie profissionais em suas atividades periciais. Tal aplicação inovaria ao preencher uma lacuna presente no rol de funcionalidades dos principais *softwares* forenses comerciais e institucionais utilizados por institutos de criminalística do país, auxiliando peritos criminais na inspeção de grandes conjuntos de arquivos de imagem.

Sob o aspecto *científico*, o presente trabalho poderá contribuir ao fornecer um estudo sobre o uso de técnicas de aprendizado de máquina para a mitigação do problema de classificação de imagens e reconhecimento de objetos no contexto da perícia criminal. Grande parte dos estudos referentes às análises de imagens em investigações digitais focam em problemas específicos como investigações de casos de pedofilia, estimação de alturas e idades de indivíduos e comprovação de adulterações em imagens, por exemplo.

Do ponto de vista *social*, espera-se que o produto obtido a partir deste estudo auxilie peritos criminais a realizarem perícias mais céleres em equipamentos de informática e dispositivos móveis. Com o processo de inspeção de imagens mais automatizado, análises serão aceleradas e, consequentemente, laudos pericias poderão ser expedidos mais rapidamente, beneficiando os órgãos solicitantes dessas demandas. Assim, espera-se que investigações criminais possam dar respostas mais rápidas à sociedade, contribuindo para o aumento da sensação de justiça e segurança da população.

## 1.6 Estrutura do trabalho

No Capítulo 2 deste trabalho, Revisão Bibliográfica, são apresentados a Teoria do Enfoque Meta-analítico Consolidado (Seção 2.1), bem como os resultados da revisão do estado da arte obtidos a partir do uso dessa teoria (Seção 2.2). Nas Seções 2.3 e 2.4 são mostradas publicações correlatas no âmbito da perícia criminal e a análise geral das publicações obtidas. A Seção 2.5 apresenta a fundamentação teórica, abrangendo conceitos importantes dentro do tema sob estudo. O Capítulo 3, Metodologia, expõe as etapas metodológicas seguidas para o atingimento dos objetivos geral e específicos descritos na Seção 1.4, tendo como base o modelo de referência CRISP-DM.

O Capítulo 4, Resultados, apresenta os resultados obtidos e expõe a análise de parte do processo de aprendizagem ocorrido durante o treinamento da rede neural convolucional. Além disso, esse capítulo mostra os critérios seguidos no desenvolvimento deste projeto para fins de sua reprodutibilidade. O Capítulo 5, Conclusões, expõe as principais conclusões obtidas a partir dos resultados do estudo, bem como apresenta e discute possibilidades de futuros trabalhos a serem desenvolvidos relacionados com o presente projeto.

# Capítulo 2

# Revisão Bibliográfica

Este Capítulo apresenta o levantamento do estado da arte, obtido por meio da aplicação da Teoria do Enfoque Meta-Analítico Consolidado (TEMAC), bem como mostra outros trabalhos correlatos desenvolvidos no âmbito da perícia criminal. A síntese de pesquisas passadas é uma das atividades mais importantes para o avanço de uma área particular, sendo o mapeamento científico um dos diversos métodos utilizados para esse processo [8]. É fornecida, também, a fundamentação teórica do presente trabalho.

## 2.1 Teoria do Enfoque Meta-Analítico Consolidado (TEMAC)

Para essa pesquisa bibliográfica, de caráter exploratório e descritivo, será utilizada a Teoria do Enfoque Meta-Analítico Consolidado (TEMAC) proposta por Mariano e Rocha [9]. A utilização do TEMAC permite, por exemplo, descobrir tendências dentro de um tema em estudo, possibilitando a descoberta de áreas em desenvolvimento e em declínio em termos de publicações e citações. O método compreende três etapas [9]:

1. preparação da pesquisa,
2. apresentação e inter-relação dos dados e
3. detalhamento, modelo integrador e validação por evidências.

Na etapa 1 são definidas palavras-chaves relacionadas ao tema objeto deste trabalho e operadores lógicos a serem fornecidos aos mecanismos de buscas de bases de dados.

Na etapa 2 do TEMAC são realizadas inter-relações entre os registros encontrados na primeira etapa, respeitando princípios e leis da bibliometria [9]. Podem ser descobertas as revistas que mais publicam sobre o tema pesquisado, os documentos mais citados no

período especificado e as conferências que mais contribuíram para o desenvolvimento da área, entre outros elementos.

Por fim, na etapa 3, são realizadas análises mais profundas sobre o tema em estudo. É possível descobrir os nomes dos autores mais relevantes na área, as principais abordagens do tema e linhas de pesquisa em expansão, por exemplo. Para o cumprimento desta etapa do TEMAC, são utilizados índices bibliométricos que detectam as relações entre autores e referências, como os índices de citações, cocitações e acoplamento.

A análise de cocitação (*co-citation*) verifica artigos que são regularmente citados juntos, sugerindo semelhanças em suas abordagens. A análise do acoplamento bibliográfico (*coupling*) possui como base a premissa de que artigos que citam trabalhos iguais, possuem similaridades [9]. A força de acoplamento de duas publicações é obtida pela quantidade de sobreposição de suas referências bibliográficas [10]. A análise de citações (*citations*) permite estimar a influência de autores, documentos e revistas tomando como base seus números de citações.

A Figura 2.1 ilustra os conceitos de acoplamento bibliográfico e análise de cocitações apresentados nesta Seção. O esquema (a) da Figura 2.1 exibe o acoplamento bibliográfico entre os artigos A e B, uma vez que essas publicações citam os mesmos artigos. O esquema (b) ilustra a análise de cocitações, pois os artigos A e B estão associados devido às citações simultâneas realizadas pelas publicações C, D e E.

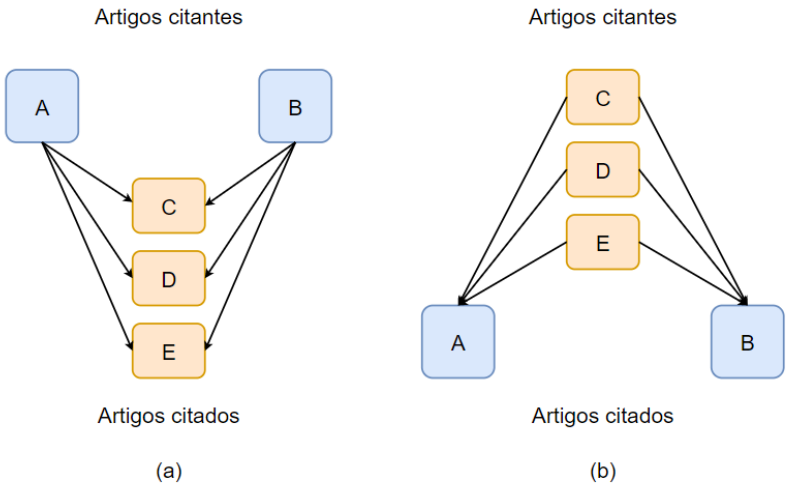



Figura 2.1: Acoplamento bibliográfico e cocitações.

Determinados termos utilizados na revisão de literatura estão em inglês a fim de manter a correlação com os campos presentes no *software* VosViwer, utilizado nesta etapa, e com os expressões pesquisadas.

## 2.2 Resultados e trabalhos correlatos obtidos por meio do TEMAC

A base de dados *Web of Science* foi o repositório escolhido para as pesquisas realizadas neste trabalho, pois constitui um dos maiores bancos de dados englobando diferentes campos de pesquisa científica [11]. As pesquisas no repositório foram efetuadas em 10/04/2020 e os artigos analisados foram os publicados entre 2011 e 2020, compreendendo um período de dez anos.

A pesquisa contemplou artigos, *preceeding papers*, *reviews* e capítulos de livros sem delimitação de áreas do conhecimento específicas, a fim de englobar uma ampla gama de aplicações dos assuntos pesquisados. Os artigos analisados estavam em inglês e deveriam possuir as expressões pesquisadas em seu título, resumo ou palavras-chaves (busca por tópico). A Tabela 2.1 apresenta as palavras-chaves e expressões escolhidas e o quantitativo de publicações científicas obtido na base de dados pesquisada.

Tabela 2.1: Quantitativo de publicações.

| **Termos** | **Resultados** |
|---|---|
| "*image classification*" | 12.800 |
| "*object detection*" OR "*object recognition*" | 25.304 |
| *forensic image analysis* | 1.869 |
| "*forensic*" AND ("*image classification*" OR "*object detection*" OR "*object recognition*") | 87 |

Analisando os resultados obtidos no repositório *Web of Science*, verificou-se que China e Estados Unidos detêm cerca de 59% da produção científica sobre classificação de imagens (*image classification*), sendo que o Brasil ocupada a 15ª posição entre os países que mais publicam acerca desse tema. O crescente número de publicações científicas verificado ao longo da última década demonstra o aumento pelo interesse na área de classificação de imagens, como apresentado na Figura 2.2.

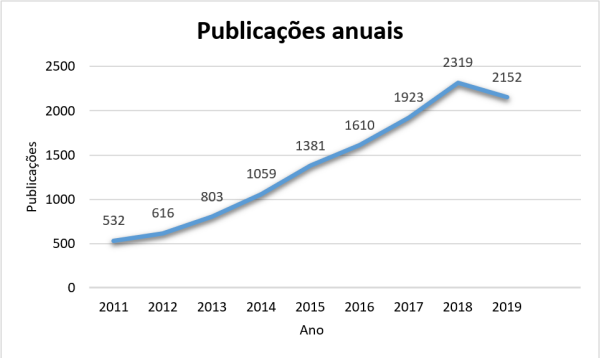



Figura 2.2: Número de publicações no últimos anos.

A fim de se observar a similaridade de abordagens entre as publicações mais citadas dentro do tema, é gerado um mapa de calor por meio do *software* livre *VosViewer* versão 1.6.15, em que os principais *clusters* possuem tonalidade avermelhada, representando trabalhos com temáticas e/ou abordagens semelhantes, conforme apresentado na Figura 2.3. Considerando o tema classificação de imagens, merecem destaque as publicações científicas mais citadas nos últimos 10 anos, as quais estão listadas na Tabela 2.2.

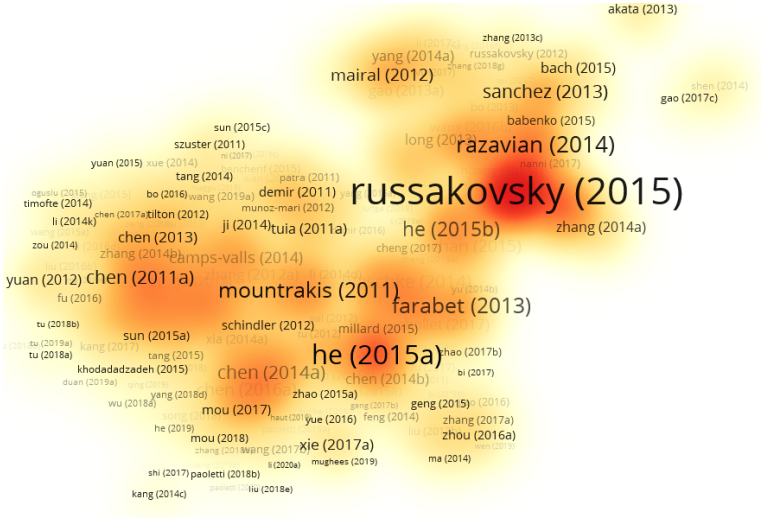



Figura 2.3: Mapa de calor para as citações (na área de classificação de imagens).

Por meio do mapa de densidade da análise de cocitações são identificados os autores que são mais frequentemente citados juntos, sugerindo semelhanças de abordagens de pesquisa. Nesse tipo de análise, diferentemente da anterior, são incluídos trabalhos publicados em um espaço de tempo mais amplo que os últimos dez anos. Pela análise das informações apresentadas na Figura 2.4, percebe-se a existência de pelo menos três grandes vertentes de estudos na área de classificação de imagens.

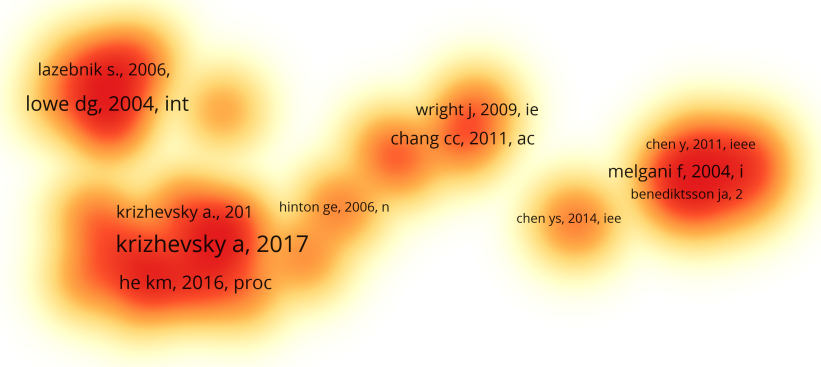



Figura 2.4: Mapa de calor para as cocitações (na área de classificação de imagens).

Tabela 2.2: Publicações mais citadas sobre classificação de imagens.

| Artigo | Autores | Ano | Citações |
|---|---|---|---|
| ImageNet Large Scale Visual Recognition Challenge | Russakovsky, Olga; Deng, Jia; Su, Hao; et al. | 2015 | 5455 |
| Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification | He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; et al. | 2015 | 2248 |
| CNN Features off-the-shelf: an Astounding Baseline for Recognition | Razavian, Ali Sharif; Azizpour, Hossein; Sullivan, Josephine; et al. | 2014 | 1258 |
| A survey on deep learning in medical image analysis classification | Litjens, Geert; Kooi, Thijs; Bejnordi, Babak Ehteshami; et al | 2017 | 1251 |
| Support vector machines in remote sensing: A review | Mountrakis, Giorgos; Im, Jungho; Ogole, Caesar | 2011 | 1201 |
| Learning Hierarchical Features for Scene Labeling | Farabet, Clement; Couprie, Camille; Najman, Laurent; et al. | 2013 | 1111 |
| Multi-column Deep Neural Networks for Image Classification | Ciresan, Dan; Meier, Ueli; Schmidhuber, Juergen | 2012 | 1073 |
| Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs | Gulshan, Varun; Peng, Lily; Coram, Marc; et al. | 2016 | 1005 |
| Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning | Shin, Hoo-Chang; Roth, Holger R.; Gao, Mingchen; et al. | 2016 | 949 |
| Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition | He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; et al. | 2015 | 921 |

Observando o mapa de calor (Figura 2.4) da análise de cocitações, nota-se maior concentração de publicações em torno de três *clusters* vermelhos, liderados pelos autores Krizhevsky, Lowe D. e Melgani F., indicando três abordagens distintas para o problema da classificação de imagens. No trabalho de Lowe D. [12] é proposto o algoritmo de visão computacional SIFT (*Scale-Invariant Feature Transform*), que permite a extração de características de imagens, para fins de identificação por meio de comparações. Esse algoritmo gera pontos-chaves na imagens, os quais são invariáveis a determinadas mudanças.

Melgani *et al* [13] em seu trabalho emprega *Support Vector Machines* (SVM) na classificação de imagens hiperespectrais. O estudo demonstrou a validade e a efetividade da

utilização de SVM como método alternativo às abordagens convencionais de reconhecimento de padrões nesse tipo de imagem. Krizhevsky *et al* [14] em seu estudo treina uma rede neural convolucional para a classificação de 1,2 milhão de imagens em alta resolução. O trabalho mostrou a capacidade desse tipo de técnica na classificação de grandes quantidades de dados, obtendo uma taxa de erro de 17% na classificação do conjunto de teste. Após o estudo das principais linhas de pesquisa, realizou-se a busca dos *fronts* de pesquisa, ou seja, das abordagens que mais estão se destacando nos últimos anos, por meio da análise de acoplamento ou *coupling* (ver Figura 2.5).

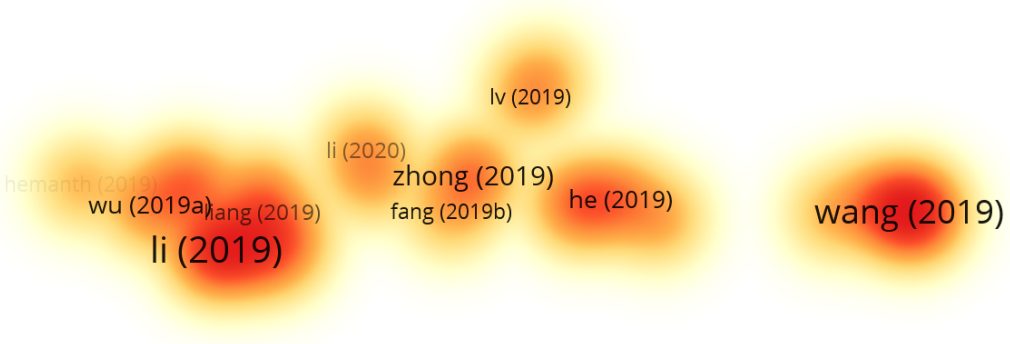



Figura 2.5: Mapa de calor para o acoplamento bibliográfico (na área de classificação de imagens).

O mapa de calor referente à análise de *coupling* exibe pelo menos dois *clusters* de tonalidade avermelhada em destaque. Eles são formados pelos autores Li (*Transfer Independently Together: A Generalized Framework for Domain Adaptation*) e Wang (*Locality and Structure Regularized Low Rank Representation for Hyperspectral Image Classification*). O trabalho de Li [15] aborda o campo de *unsupervised heterogeneous domain adaptation*, enquanto o Wang [16] estuda o processo de classificação de imagens hiperespectrais.

Além dessa análise, foram extraídas e agrupadas todas as palavras presentes em títulos e resumos de artigos científicos publicados nos últimos dez anos acerca da classificação de imagens (*image classification*). Verificaram-se que existem dois agrupamentos em evidência: um grupo liderado pelos termos *support vector machine* e outro liderado pela expressão *convolutional neural networks*. Por essa análise, constata-se uma migração na abordagem de classificação de imagens: as primeiras publicações do período analisado empregavam algoritmos de *Support Vector Machines* (SVM), ao passo que nas publicações mais recentes há um emprego maior de técnicas de *Deep learning*, como as *Convolutional Neural Networks*.

Assim, após a realização das três etapas do TEMAC para a expressão "*image classification*", em que se realizaram as análises de citações, de cocitações, de acoplamento bibliográfico e de mapeamento textual de títulos e resumos, selecionou-se o conjunto de

publicações mais citadas. Esses artigos foram estudados e suas principais contribuições na área de classificação de imagens estão presentes na Tabela 2.3.

Tabela 2.3: Enfoques das publicações mais citadas na área de classificação de imagens.

| Artigo | Ano | Enfoque |
|---|---|---|
| ImageNet Large Scale Visual Recognition Challenge | 2015 | O artigo descreve a criação do conjunto de dados do benchmark ImageNet Large Scale Visual Recognition Challenge e os avanços em reconhecimento de objetos obtidos a partir dessa iniciativa. |
| Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification | 2015 | Estuda o emprego de rectifier neural networks para a classificação de imagens sob dois aspectos. É proposto o Parametric Rectified Linear Unit (PReLU) que generaliza a tradicional unidade retificadora, sendo derivado de um robusto método de inicialização que considera as não-linearidades do retificador. |
| CNN Features off-the-shelf: an Astounding Baseline for Recognition | 2014 | Utiliza uma Convolutional Neural Networks (CNN) off-the-shelf com classificadores simples na tarefa de reconhecimento. O modelo CNN foi originalmente otimizado para o desafio de classificação de objetos da base do ILSVRC 2013. O autor conclui que deep learning com CNN deve ser considerado como primeiro candidato em qualquer tarefa de reconhecimento visual. |
| A survey on deep learning in medical image analysis classification | 2017 | Apresenta um estudo contendo os principais conceitos de Deep Learning aplicáveis à tarefa de classificação de imagens na área médica. Além disso, são apresentados, de forma resumida, trabalhos desenvolvidos nesse campo. Também é apresentado o atual estado-da-arte na classificação de imagens médicas, além de discutir os principais desafios e oportunidades na área, para pesquisas futuras. |
| Support vector machines in remote sensing: A review | 2011 | Apresenta uma revisão de publicações que tratam do uso de Support Vector Machines (SVM) em sensoriamento remoto. |
| Learning Hierarchical Features for Scene Labeling | 2013 | Propõe um método que usa CNN multiscale treinada na tarefa de identificação da categoria a qual pertence cada região de uma imagem. A CNN feed-forward apresentada obteve performance comparável à produzida por algoritmos do estado-da-arte. |
| Multi-column Deep Neural Networks for Image Classification | 2012 | Emprega Deep Convolutional Neural Networks (DNN) na classificação de imagens. A combinação de algumas colunas da DNN em uma estrutura DNN multi-coluna (MCDNN) propiciou queda na taxa de erros na ordem de 30 a 40%. Apresenta resultados superiores ao estado-da-arte até então vigente na tarefa de classificação de imagens. |
| Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs | 2016 | Aplica deep learning para a classificação automática de diabetes a partir de fotografias da retina. Os modelos apresentaram alta taxa de sensibilidade e especificidade na detecção de diabetes. |
| Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning | 2016 | Neste estudo são avaliadas diferentes arquiteturas de Convolutional Neural Networks (CNN), bem como é examinado quando e como transfer learning, a partir do conjunto ImageNet, pode ser útil. |
| Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition | 2015 | Implementa uma estratégia para contornar a limitação no tamanho para as imagens de entrada em uma CNN. Eliminando essa limitação de tamanho, o modelo foi capaz de obter resultados superiores na tarefa de detecção de objetos. |

Prosseguindo a revisão do estado da arte, foi realizada a pesquisa na base de dados *Web of Science* para a expressão "*object detection*" OR "*object recognition*", também de interesse deste projeto, seguindo os mesmos passos do TEMAC abordados para o termo anterior. Estados Unidos e China são os principais países desenvolvedores de pesquisas científicas na área, compreendendo cerca de 49% das publicações sobre o tema. O Brasil ocupa a 13ª colocação entre os países com mais publicações acerca de detec-

ção/reconhecimento de objetos. Essa área apresenta, ao longo dos últimos dez anos, aumento no número de publicações científicas, conforme apresenta a Figura 2.6.

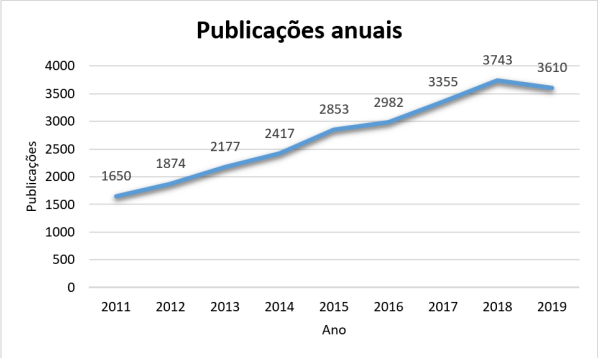



Figura 2.6: Quantitativo de publicações sobre detecção/reconhecimento de objetos nos últimos dez anos.

A Tabela 2.4 apresenta a lista de artigos científicos mais citados publicados nos últimos dez anos sobre o tema *object detection/recognition*.

Tabela 2.4: Dez publicações científicas mais citadas relacionadas a detecção/reconhecimento de objetos.

| **Artigo** | **Autores** | **Ano** | **Citações** |
|---|---|---|---|
| Deep learning | LeCun, Yann; Bengio, Yoshua; Hinton, Geoffrey | 2015 | 12029 |
| Deep Residual Learning for Image Recognition | He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; et al. | 2016 | 11066 |
| ImageNet Large Scale Visual Recognition Challenge | Russakovsky, Olga; Deng, Jia; Su, Hao; et al. | 2015 | 5455 |
| Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks | Ren, Shaoqing; He, Kaiming; Girshick, Ross; et al. | 2017 | 5257 |
| Rich feature hierarchies for accurate object detection and semantic segmentation | Girshick, Ross; Donahue, Jeff; Darrell, Trevor; et al. | 2014 | 3938 |
| Deep learning in neural networks: An overview | Schmidhuber, Juergen | 2015 | 3605 |
| Representation Learning: A Review and New Perspectives | Bengio, Yoshua; Courville, Aaron; Vincent, Pascal | 2013 | 2670 |
| Fast R-CNN | Girshick, Ross | 2015 | 2662 |
| ORB: an efficient alternative to SIFT or SURF | Rublee, Ethan; Rabaud, Vincent; Konolige, Kurt; et al. | 2011 | 2444 |
| You Only Look Once: Unified, Real-Time Object Detection | Redmon, Joseph; Divvala, Santosh; Girshick, Ross; et al. | 2016 | 2179 |

A partir do mapa de calor de citações, obtido por meio do software *VosViewer* versão 1.6.15, verifica-se a formação de agrupamentos de publicações contendo similaridades nas abordagens do problema de detecção/reconhecimento de objetos. Áreas de tonalidades avermelhadas no mapa constante na Figura 2.7 indicam maior quantidade de publicações que adotam uma abordagem específica.

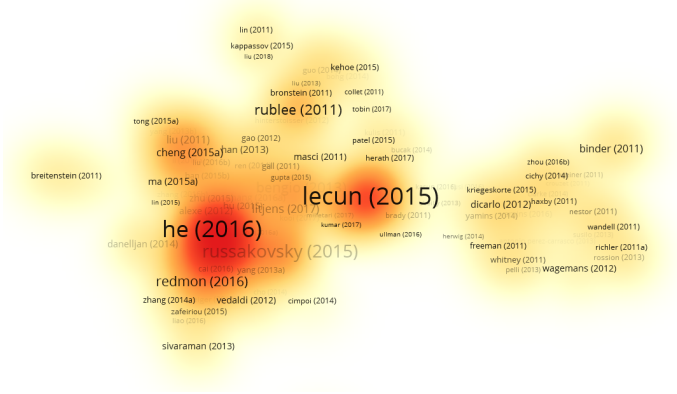



Figura 2.7: Mapa de calor para as citações dos artigos (na área de detecção/reconhecimento de objetos).

O mapa de calor de citações apresenta dois *clusters* principais e distantes, indicando enfoques distintos na abordagem do problema sobre identificação/reconhecimento de objetos. O trabalho de Lecun [17], principal publicação do primeiro *cluster*„„, aborda o tema *Deep Learning*, apresentando suas principais características, aplicações e desafios. O trabalho de He *et al* [18], publicação de destaque no segundo *cluster*, apresenta um *framework* para facilitar o treinamento de redes nerais.

Por meio da análise de cocitações, que parte da premissa de que artigos que são citados juntos tendem a possuir similaridades, é gerado o mapa de calor ilustrada na Figura 2.8. A exemplo do mapa para as citações, no caso das cocitações também observa-se a presença de dois *clusters* em destaque. O aglomerado liderado pelo artigo de Lowe [12], o qual aborda o algoritmo de visão computacional SIFT (*Scale-Invariant Feature Transform*), citado anteriormente, é o que possui relação mais próxima ao discutido neste trabalho.

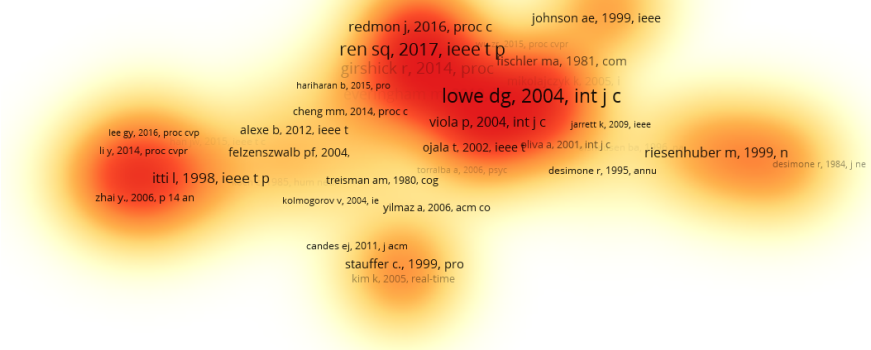



Figura 2.8: Mapa de calor para as cocitações (na área de detecção/reconhecimento de objetos).

A Figura 2.9, gerada a partir do site *WordItOut* [19], apresenta o mapa de palavras contendo as palavras-chaves mais comuns presentes em publicações científicas, dos anos de 2019 e 2020, sobre o tema detecção/reconhecimento de objetos. As palavras mais citadas são *learning, deep, memory, model, radar, convolutional, neural, network, feature* e *segmentation*.

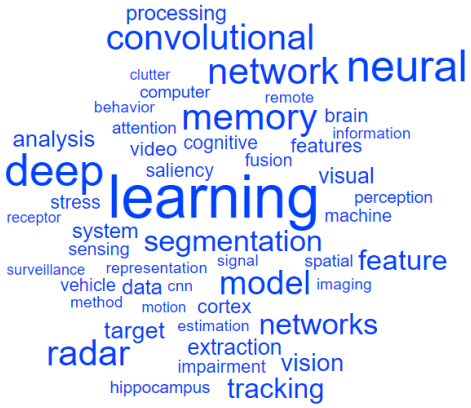



Figura 2.9: Principais palavras-chaves presentes nos artigos de 2019 e 2020 (na área de detecção/reconhecimento de objetos).

A fim de conhecer as principais frentes de pesquisas atuais, foi realizada a análise de acoplamento bibliográfico. O mapa de calor da Figura 2.10 mostra, de forma proeminente, os trabalhos de Zhao *et al* [20] e Lourenco *et al* [21]. O primeiro artigo apresenta uma revisão de trabalhos que empregam técnicas de *deep learning* na detecção de objetos, enquanto que a publicação de Lourenco *et al* investiga os efeitos da representação da forma no reconhecimento de objetos.


Figura 2.10: Mapa de calor para o acoplamento bibliográfico (na área de detecção/ reconhecimento de objetos).

Por meio das pesquisas e análises das publicações científicas acerca dos termos escolhidos, presentes na base de dados da *Web of Science*, identificaram-se trabalhos de grande relevância e tendências de abordagens acerca do tema. A Tabela 2.5 exibe, de forma resumida, os enfoques dos trabalhos mais citados na área de detecção/reconhecimento de objetos.

Tabela 2.5: Enfoques das publicações mais citadas na área de detecção/reconhecimento de objetos.

| Artigo | Ano | Enfoque |
|---|---|---|
| Deep Learning | 2015 | Apresenta uma revisão das principais técnicas de Deep learning como CNN e Recurrent Neural Network (RNN), bem como analisa as perspectivas de utilização de deep learning. |
| Deep Residual Learning for Image Recognition | 2016 | Apresenta um framework para facilitar o treinamento de redes nerais. As camadas são reformuladas como funções de aprendizado residual, ao contrário das funções anteriormente usadas. São exibidos resultados da classificação do banco de imagens ImageNet. |
| ImageNet Large Scale Visual Recognition Challenge | 2015 | O artigo descreve a criação do conjunto de dados do benchmark ImageNet Large Scale Visual Recognition Challenge e os avanços em reconhecimento de objetos obtidos a partir dessa iniciativa. |
| Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks | 2017 | Apresenta o Region Proprosal Network (RPN) para detecção de objetos em tempo real. |
| Rich feature hierarchies for accurate object detection and semantic segmentation | 2014 | É proposto um algoritmo de detecção que obteve mean avearege precision (mAP) 30% superior ao melhor colocado no desafio VOC 2012. |
| Deep learning in neural networks: An overview | 2015 | Apresenta uma revisão das principais publicações da área de deep learning. |
| Representation Learning: A Review and New Perspectives | 2013 | Apresenta uma revisão dos principais trabalhos nas áreas de unsupervised feature learning e deep learning. |
| Fast R-CNN | 2015 | Propõe o método Fast Region-based Convolutional Network (Fast R-CNN) para detecção de objetos. |
| ORB: an efficient alternative to SIFT or SURF | 2011 | Apresenta um descritor binário rápido baseado em BRIEF, chamado ORB. Por meio de experimento ORB apresentou velocidade superiores ao SIFT. A eficiência foi testado em aplicações reais, incluindo detecção de objetos e patch-tracking utilizando um smartphone. |
| You Only Look Once: Unified, Real-Time Object Detection | 2016 | Apresenta a abordagem YOLO para a detecção de objetos, a qual superou outros métodos de detecção de objetos como DPM e R-CNN, quando generalizado de imagens naturais para outros domínios como o artístico. |

Prosseguindo a revisão do estado da arte realizada com base no TEMAC foram efetuadas pesquisas no repositório *Web of Science* para o expressão *forensic image analysis*. Das 1.869 publicações científicas realizadas entre os anos de 2011 e 2020, cerca de 30% estão relacionadas à área de medicina legal. Estados Unidos e China são os países que mais contribuíram com publicações na área, totalizando cerca de 28% da produção total. O Brasil ocupa a 9ª posição entre as nações com maior quantidade de publicações, porém a Universidade Estadual de Campinas (Unicamp) é a organização com maior quantidade de artigos vinculados, com cerca de 1,9% do total disponível na base de dados.

O mapa de calor referente às citações dos artigos publicados nos últimos dez anos para o tema *forensic image analysis*, presente na Figura 2.11, destaca dois enfoques distintos para o tema. Os trabalhos de maior destaque, de Amerini *et al* [22] e Huang *et al* [23], abordam um problema recorrente na área de perícias de arquivos de imagem: a detecção

de possíveis adulterações em imagens, especialmente nos casos em uma região é copiada e colada em outra área.

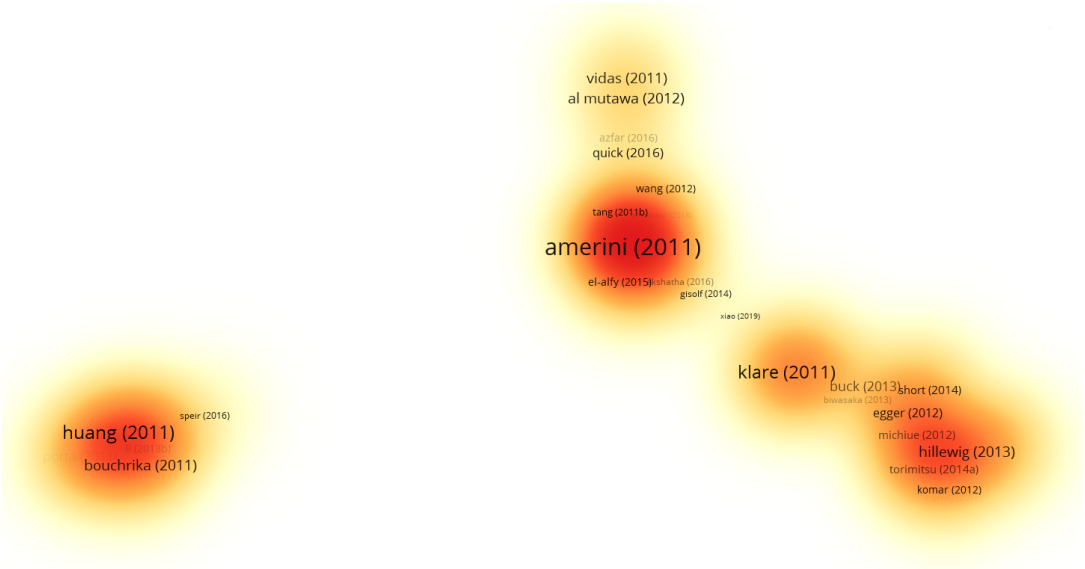



Figura 2.11: Mapa de calor para as citações de artigos publicados nos últimos dez anos sobre o tema *forensic image analysis*.

A Figura 2.12 exibe a nuvem contendo os termos mais citados presentes nas palavras-chaves dos artigos analisados. Destacam-se as palavras *forensics*, *identification*, *age*, *detection*, *estimation*, *digital*, *forgery*, *anthropology*, *tomography*, entre outras, indicando as principais áreas e atividades envolvidas no que diz respeito à análise de imagens na esfera pericial.


Figura 2.12: Nuvem de palavras com as principais palavras-chaves presentes nas publicações (na área de *forensic image analysis*).

Por fim, realizou-se a seleção de trabalhos científicos, publicados entre os anos de 2011 e 2020 e disponibilizados no repositório *Web of Science*, relacionados à classificação de imagens e detecção/reconhecimento de objetos na atividade pericial. Para isso, foi

fornecida a expressão lógica *(forensic* AND ("image classification" OR "object detection" OR "object recognition"))* ao mecanismo de busca do WoS, o qual retornou 87 publicações científicas.

Tabela 2.6: Publicações científicas mais citadas sobre o tema pesquisado.

| **Artigo** | **Autores** | **Ano** | **Citações** |
|---|---|---|---|
| Demographic Estimation from Face Images: Human vs. Machine Performance | Han, Hu; Otto, Charles; Liu, Xiaoming; et al. | 2015 | 124 |
| A Kernel-Based Feature Selection Method for SVM With RBF Kernel for Hyperspectral Image Classification | Kuo, Bor-Chen; Ho, Hsin-Hua; Li, Cheng-Hsuan; et al. | 2014 | 119 |
| Survey on blind image forgery detection | Qazi, Tanzeela; Hayat, Khizar; Khan, Samee U.; et al. | 2013 | 47 |
| Revealing the Trace of High-Quality JPEG Compression Through Quantization Noise Analysis | Li, Bin; Ng, Tian-Tsong; Li, Xiaolong; et al. | 2015 | 24 |
| Image Origin Classification Based on Social Network Provenance | Caldelli, Roberto; Becarelli, Rudy; Amerini, Irene | 2017 | 21 |
| Detection of object-based manipulation by the statistical features of object contour | Chen Richao; Yang Gaobo; Zhu Ningbo | 2014 | 19 |
| Fusion of block and keypoints based approaches for effective copy-move image forgery detection | Zheng, Jiangbin; Liu, Yanan; Ren, Jinchang; et al. | 2015 | 16 |
| 3D Fragment Reassembly using Integrated Template Guidance and Fracture-Region Matching | Zhang, Kang; Yu, Wuyi; Manhein, Mary; et al. | 2015 | 16 |
| Bimodal Biometric System based on SIFT Descriptors of Hand Images | Charfi, Nesrine; Trichili, Hanene; Alimi, Adel M. et al. | 2014 | 13 |
| Novel hand biometric system using invariant descriptorss | Charfi, Nesrine; Trichili, Hanene; Alimi, Adel M.; et al. | 2014 | 9 |

A Tabela 2.6 apresenta a lista dos artigos mais citados obtidos a partir da expressão pesquisada para o período de interesse. A nuvem de palavras, exibida na Figura 2.13, contém as palavras-chaves mais comuns presentes nos artigos acerca do tema. A partir da lista de publicações mais citadas e das palavras-chaves mais comuns, conclui-se que parte

significativa das publicações que tratam da análise de imagens no ambiente da perícia digital estão relacionadas à investigação de adulteração de arquivos de imagem e vídeo.

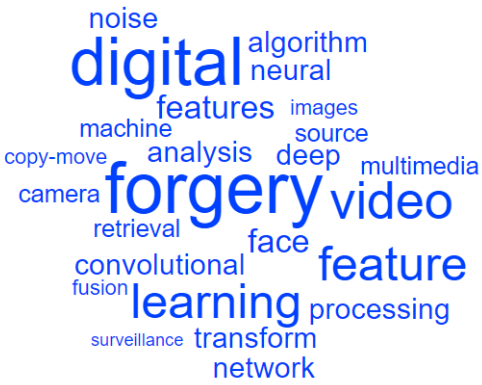



Figura 2.13: Nuvem de palavras-chaves das publicações científicas.

## 2.3 Outros trabalhos correlatos

Além de trabalhos selecionados utilizando-se o TEMAC, destacam-se, também, estudos publicados no Brasil e no exterior que utilizam técnicas de inteligência artificial nas mais diversas áreas da perícia criminal. A seguir são exemplificados estudos aplicados nos campos da medicina legal, documentoscopia e grafoscopia, e destacadas as principais contribuições de projetos na computação forense.

### 2.3.1 Medicina legal e perícias em locais de crimes contra pessoa

Na área de medicina legal há estudos recentes que envolvem, por exemplo, a determinação da idade óssea de uma pessoa a partir de imagens de raio-x [24] e a estimação do sexo a partir de radiografias de estruturas faciais [25]. Com relação a projetos que empregam análise de imagens a fim de auxiliar investigações de cenas de crimes há, por exemplo, o estudo que emprega transferência de aprendizado (*transfer learning*) na tarefa de classificação de arquivos de imagem, sendo capaz de identificar determinados elementos de interesse pericial, como portas, janelas e manchas de sangue em fotos relacionadas a locais de crimes diversos [26].

### 2.3.2 Balística, documentoscopia e grafoscopia

No que tange à aplicação de técnicas de *machine learning* no campo da balística forense, pode-se citar o recente trabalho que visa identificar uma arma de fogo por meio da análise dos ruídos produzidos por seus mecanismos [27], demonstrando a aplicabilidade de algoritmos como *Linear Discriminant Analysis* (LDA) e *Extreme Gradient Boosting*

(XGBoost) nessa classificação. Em documentoscopia e grafoscopia também há estudos relacionando aprendizado de máquina e soluções de problemas específicos nesses ramos da perícia criminal. Em [28] é exemplificada a identificação de impressoras a partir da análise de documentos impressos por esses equipamentos. O estudo [29] mostra uma extensa revisão dos principais métodos de identificação do escritor, por meio de seus registros manuscritos. São apresentados artigos científicos que relatam a utilização de diversas técnicas nessa atividade, como *K-nearest neighbor* (KNN), *Support Vector Machine* (SVM) e *deep learning*.

### 2.3.3 Computação forense

Com relação ao uso de técnicas de *machine learning* em computação forense, observa-se grande quantidade de estudos que visam identificar e analisar o comportamento de *softwares* e comandos maliciosos (que exploram vulnerabilidades dos sistemas a fim de executar ações indesejadas) em diversas plataformas. O trabalho [30] explora a detecção de *malwares* na plataforma Android a partir da conversão de arquivos binários suspeitos em imagens e posterior classificação com redes neurais convolucionais. Na plataforma Windows, merecem destaque estudos recentes que envolvem a detecção de comandos maliciosos no *PowerShell* utilizando-se algoritmos de *Deep Neural Networks* [31], e o estudo [32] que emprega diferentes algoritmos na análise do registro do Windows com o intuito de identificar a possível presença *malware*, além de indicar o seu tipo.

Como outros projetos que relacionam o emprego de algoritmos de aprendizado de máquina na perícia computacional forense, destacam-se o uso de CNN na identificação do tipo de arquivo a partir de seus fragmentos [33], a detecção de possíveis alterações no *timestamp* de arquivos em sistemas NTFS [34], e a identificação de sistemas operacionais por meio da análise de pacotes de dados enviados por uma rede. Nesse processo de identificação, os algoritmos KNN, árvores de decisão e redes neurais artificiais foram implementados e comparados, apresentando o último algoritmo citado a maior taxa de acurácia [35]. Na análise de esteganografia, técnica que consiste no ocultamento de uma mensagem dentro de arquivos, conhecida como *steganalysis*, há uma ampla gama de estudos que utilizam algoritmos para detecção de elementos ocultos em arquivos de texto, áudio, vídeo e imagem, como os apresentados em [36], [37] e [38].

Devido ao largo emprego de arquivos de imagem na comunicação, diversos trabalhos estão sendo desenvolvidos com o objetivo de auxiliar os peritos criminais na investigação desse tipo de arquivo. Detecção de manipulação em imagens [39], [26] e [40], identificação da câmera fotográfica a partir da análise de suas fotografias geradas [41] e [42], identificação de elementos de interesse em imagens, como roupas [43] e faces humanas [44], são exemplos que evidenciam a importância da análise de imagens na perícia criminal.

Em virtude da relevância e da maior proximidade de determinados estudos e o presente projeto, serão apresentados as principais contribuições de sete trabalhos desenvolvidos no país e no exterior. Em [45], desenvolvido por peritos criminais da Polícia Federal, é apresentada uma ferramenta para auxiliar a perícia computacional na detecção de arquivos de mídia contendo potenciais imagens de pornografia infantil. São realizadas análises nos conteúdos de arquivos de imagem e vídeos, bem como análises de nomes de arquivos e *hash*. Em arquivos de imagem, por exemplo, são aplicadas técnicas de detecção de cor de pele para detectar nudez humana. O trabalho contribui ao auxiliar a detecção de possível pornografia infantil em dispositivos objeto de exames periciais, porém é uma ferramenta específica para a investigação desse único tipo de crime.

Em George [46] o autor apresenta determinadas aplicações de algoritmos de mineração de dados na atividade pericial. Utilização de regras de associação para extração de informações relevantes em arquivos de *log*, análise de *outliers* para detecção de pastas ocultas, e *Support Vector Machines* (SVM) para determinação de autoria de e-mails são algumas das aplicações citadas no estudo. O estudo demonstra o emprego de algoritmos de mineração de dados para a determinação de propriedade e categorização por tipo de arquivos.

No trabalho Perez [47] são aplicados algoritmos de *deep learning* combinado com análise estática (imagem) e dinâmica (movimento) do conteúdo de arquivos para identificar possíveis conteúdos pornográficos. O estudo demonstrou, principalmente, a eficácia do emprego conjunto de algoritmos de *deep learning* com a análise de movimento para detecção de pornografia, aumentado os índices de detecção se comparado a ferramentas tradicionais, as quais utilizam, muitas vezes, apenas análise de cor de pele.

No estudo de Marturana [48] é explorado o uso de algoritmos de aprendizado de máquina no processo de triagem de materiais, fase inicial da perícia computacional, ao automatizar a categorização de dispositivos com base em seu conteúdo e tipo de crime sob investigação. Para classificar os equipamentos que serão objeto de perícia é proposta a utilização de determinados parâmetros na análise como número de contatos, ligações realizadas e recebidas, mensagens enviadas e recebidas e quantidade de imagens e vídeos baixados. Apesar de fornecer resultados satisfatórios, o estudo não investiga a aplicação dos algoritmos na classificação de imagens baseada em seus conteúdos, potencialmente relacionados ao ato criminal sob investigação.

No trabalho de Saikia [49] é apresentado um sistema de tempo real para detecção de objetos em ambientes internos, utilizando algoritmos de *deep learning*. O sistema desenvolvido detecta objetos típicos presentes em quartos, auxiliando peritos na análise de cenas de crimes. A ferramenta mostrou-se eficaz para a detecção de elementos em cenas de crimes reais, porém o estudo não abarca a aplicação de técnicas em imagens já

coletadas e possivelmente relacionados a locais de crime.

Em Mayer [50] é proposto o emprego de algoritmos de *deep learning* a fim de agrupar imagens facilitando a detecção de conteúdo pornográfico. O estudo propiciou a classificação satisfatória de imagens, acelerando a descoberta de imagens com conteúdo pornográfico. A exemplo da maioria dos trabalhos presentes na área, foi focada apenas a utilização de algoritmos de *deep learning* na tarefa específica de detecção de nudez, não permitindo o seu emprego em perícias onde o reconhecimento de outros elementos visuais é necessário.

No trabalho de Nassif [51] é exibido um estudo que analisa a eficácia do emprego de algoritmos de agrupamento na classificação de documentos, unicamente textuais, objeto de perícia na Polícia Federal. O estudo exibe o potencial de aplicações de clusterização em bases de dados reais para acelerar o processo de inspeção de arquivos em computadores.

## 2.4 Análise dos trabalhos apresentados

A utilização do TEMAC possibilitou obter relevantes informações relativas a aplicações e tendências nas áreas de classificação de imagens e detecção de objetos, como apresentado na Seção 2.2. A Seção seguinte exemplificou trabalhos na área da perícia criminal que empregam técnicas de *machine learning*, evidenciando sua vasta aplicabilidade para a solução de problemas diversos.

No que diz respeito à área da computação forense, observou-se que a maioria dos trabalhos focam na análise de esteganografia, na detecção de possíveis manipulações em imagens e vídeos ou na identificação de imagens com possível conteúdo pedo-pornográfico. Assim, verifica-se que a maioria dos trabalhos não aborda a questão da triagem de imagens por meio de uma classificação prévia e a aplicação de reconhecimento de objetos que podem estar relacionados a uma ampla gama de crimes.

## 2.5 Fundamentação teórica

Esta Seção apresenta os principais conceitos relacionados ao desenvolvimento do presente projeto.

### 2.5.1 Modelo de referência CRISP-DM

Com as tecnologias atuais é possível realizar a coleta de uma grande quantidade de dados, das mais diversas fontes. Desse modo, o principal problema não está na descoberta e coleta de dados, mas na análise e extração de conhecimento a partir desse grande conjunto

de informações [52]. Assim, surge o conceito de *data mining* (mineração de dados) que, por meio de técnicas específicas, visa obter conhecimento por meio da descoberta e extração de importantes padrões em dados, podendo contribuir para estratégias de empresas, construções de bases de conhecimentos, pesquisas médicas e científicas [53].

A mineração de dados necessita de uma abordagem padronizada que auxiliará na tradução dos problemas de negócio em tarefas de *data mining*, sugerindo as apropriadas transformações nos dados e técnicas de mineração [54]. Nesse cenário, surgiram diferentes *frameworks*, como o CRISP-DM (*Cross Industry Standard Process for Data Mining*), para lidar com o processo de *data mining*.

CRISP-DM é um modelo de referência, que consiste de seis fases, formado por um processo iterativo que começa com o entendimento do negócio. Após essa etapa inicial, é realizado o entendimento dos dados e o seu processamento para a etapa de modelagem, em que algoritmos de *machine learning* podem ser aplicados. Os resultados obtidos pelo modelo são avaliados e, se adequados para o cenário estudado, são implementados. As seis etapas do ciclo de CRISP-DM estão apresentadas na Figura 2.14 e descritas a seguir [54]:

1. Entendimento do negócio (*Business understading*): foca no entendimento dos objetivos e necessidades do projeto sob uma perspectiva de negócio, convertendo, então, esse conhecimento em uma definição de um problema de *data mining* e um projeto preliminar que visa atingir esses objetivos.

2. Entendimento dos dados (*Data understanding*): inicia-se com a coleta de dados e envolve uma série de atividades como a identificação de problemas de qualidade nos dados, descoberta dos primeiros *insights* acerca dos dados ou detecção de subconjuntos de interesse.

3. Preparação dos dados (*Data preparation*): essa fase compreende a construção do *dataset* final (dados que serão fornecidos para as ferramentas de modelagem) a partir dos dados brutos iniciais. Essa etapa inclui seleção de tabelas, registros e atributos, limpeza dos dados, geração de novos atributos e transformações dos dados.

4. Modelagem (*Modeling*): nesta fase são selecionadas e aplicadas várias técnicas de modelagem, e os seus parâmetros são ajustados para a obtenção dos melhores valores.

5. Avaliação (*Evaluation*): consiste na avaliação do modelo e na revisão das etapas executadas em seu desenvolvimento. Um objetivo chave dessa etapa é determinar se alguma questão importante de negócio não foi adequadamente verificada.

6. Implementação (*Deployment*): o conhecimento obtido deve ser organizado e fornecido no ambiente real para ser usado na prática. Essa implementação pode se dar de forma simples, como a geração de relatórios, ou complexas, como a implementação de um processo de *data mining*.

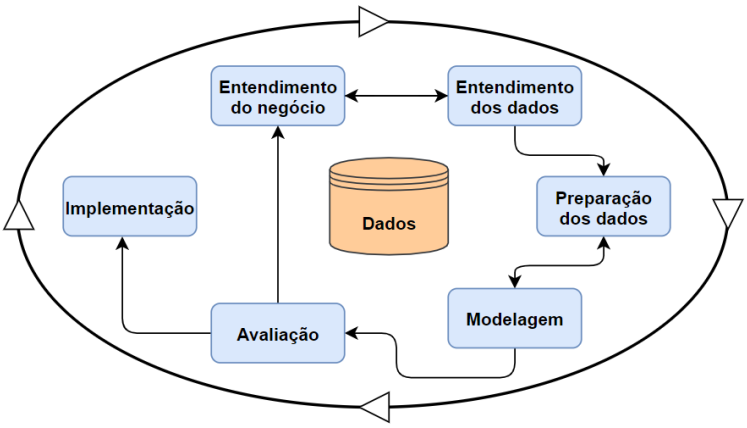



Figura 2.14: Fases dos CRISP-DM.

### 2.5.2 Processamento digital de imagens

Uma imagem pode ser conceituada como uma função *f(x,y)*, em que $x$ e $y$ são coordenadas espaciais (plano) e a amplitude de $f$ em qualquer par de coordenadas *(x,y)* pode ser chamada de *intensidade* ou *nível de cinza da imagem* para esse ponto. Uma imagem digital é aquela em que os valores de $x$, $y$ e a intensidade de $f$ são valores finitos e discretos. Nesse contexto, o processamento digital de imagens refere-se ao processamento de imagens digitais realizadas por um computador digital [55].

Uma imagem digital é constituída por um número finito de elementos, com localização e valor específicos. Esses elementos constituintes da imagem são chamados *elementos pictóricos, elementos de imagem, pels* e *pixels*, sendo esse último o mais utilizado para representá-la [31]. Uma imagem pode ser representada por uma matriz bidimensional, em que cada elemento dessa matriz corresponde, então, a um *pixel*. Assim, a posição de um *pixel* em uma imagem pode ser expressa utilizando-se a notação de matrizes, por meio de índices indicadores de sua linha e coluna [56], conforme disposto na Figura 2.15.

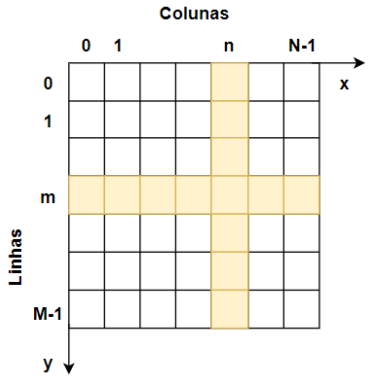



Figura 2.15: Representação de uma imagem digital por meio de vetores de pontos discretos em uma matriz bidimensional.

Ao se observar a Figura 2.15, verifica-se que a origem da imagem está situada no extremo superior esquerdo, estendendo-se para a direção direita e para baixo. Trata-se de uma representação convencional de imagem, uma vez que diversos dispositivos de visualização percorrem uma imagem partindo-se do canto superior esquerdo em direção à direita, uma linha por vez [55]. O conceito de *resolução espacial* está relacionado ao menor detalhe discernível em uma imagem, podendo ser expressa, dentre outras formas, pelo número de *pixels* por unidade de distância, sendo essa medida bastante utilizada por editoras e indústrias gráficas [55].

Uma *imagem colorida* é uma imagem multibanda, em que a cor em cada ponto *(x,y)* é definido pelo seu brilho, matiz e saturação. O brilho incorpora a noção acromática de intensidade, a matiz diz respeito ao comprimento de onda dominante em uma mistura de ondas de luz, e a saturação se refere à pureza relativa ou à quantidade de luz branca misturada a uma matiz [55]. A maioria das cores visíveis pelo olho humano pode ser representada como uma combinação das cores primárias vermelho (R), verde (G) e azul (B). Desse modo, uma representação comum para uma imagem colorida emprega três bandas (R, G e B) com profundidade de 1 *byte* por *pixel* [57]. Considerando-se esse tamanho por *pixel*, tem-se valores entre 0 e 255 para cada um dos três canais, os quais são combinados a fim de produzir o conjunto de cores da imagem.

No modelo RGB, o subespaço de cores de interesse é constituído de um cubo (espaço tridimensional), conforme apresentado na Figura 2.16 , em que os valores RGB primários estão situados nos três vértices, as cores secundárias ciano, magenta e amarelo estão em outros três vértices, o preto está na origem e o branco é o vértice mais distante da origem. Conforme observado na imagem, a escala de cinza (em que a intensidade de cada *pixel* pode variar de 0 a 255) estende-se do preto até o branco, percorrendo o segmento de reta que liga esses dois pontos. As diferentes cores nesse modelo são pontos no cubo ou dentro dele e são definidas por vetores que se estendem a partir da origem.

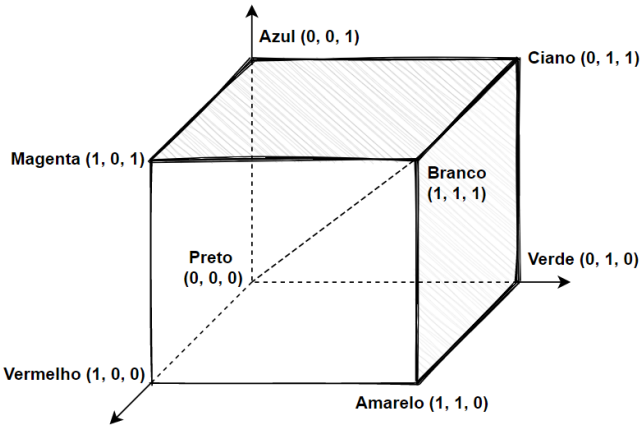



Figura 2.16: Espaço de cores RGB.

Há dois tipos básicos de imagens digitais: imagens *raster* (mapa de *bits* ou *bitmap*) e imagens vetoriais (*vector graphics*). As imagens do tipo *raster* são constituídas por *pixels* dispostos em um arranjo matricial utilizando coordenadas discretas. As imagens vetoriais representam objetos geométricos empregando coordenadas contínuas, que são somente rasterizadas quando são exibidas em um dispositivo físico como um monitor ou impressora [58].

Como exemplo de formatos de imagens vetoriais destacam-se CGM (*Computer Graphics Metafile*), SVG (*Scalable Vector Graphics*), DXF (*Drawing Exchange Format* da Auto-Desk), AI (*Adobe Illustrator*), PICT (*QuickDrw Graphics Metafile* da Apple) e WMF/EMF (*Windows Metafile* e *Enhanced Metafile from Windows*) [58]. A seguir serão apresentados os formatos de arquivos de imagem digital mais comuns:

- *Tagged Image File Format (TIFF)*: formato bastante flexível que suporta vários tipos de compressão e é capaz de comportar imagens e dados. A sua capacidade de armazenar imagens em um formato com compressão sem perda de dados o torna bastante útil, pois permite a edição e salvamento sem perda de qualidade.

- *Graphics Interchange Format (GIF)*: formato de arquivo que suporta a compressão sem perda de dados do tipo LZW para codificar de modo eficiente grandes áreas com a mesma cor.

- *Portable Network Graphics (PNG)*: esse formato foi desenvolvido originalmente para substituir o GIF em virtude de questões de licenciamento. Foi desenvolvido para ser o formato de imagem universal para uso na internet. Como diferenças para o formato GIF, PNG apresenta um canal para transparência, porém não permite incluir múltiplas imagens em um único arquivo, não fornecendo, assim, animações como as presentes em GIF [58].

- *Joint Picture Experts Group (JPG ou JPEG)*: é o formato de arquivo de imagem mais utilizado atualmente [58], permitindo uma alta taxa de compressão mantendo a qualidade da imagem. Como ponto negativo, o algoritmo de compressão do JPEG apresenta baixa performance em imagens não fotográficas, como gravuras (foi concebido para trabalhar com fotografias) [58].

- *Exchangeable Image File Format (EXIF)*: é uma variação do formato JPEG desenvolvido para armazenar imagens obtidas de câmeras digitais. Metadados importantes da imagem podem ser armazenados, como tipo de câmera, data e hora, parâmetros utilizados na fotografia e localização espacial [58].

- *Windows Bitmap (BMP)*: formato comum na plataforma Windows, suportando imagens com 1, 4, 8, 16, 24 e 32 *bits* por *pixel* [59].

### 2.5.3 Reconhecimento de padrões

O reconhecimento de padrões pode ser conceituado como a classificação de dados baseada no conhecimento adquirido ou em informações estatísticas extraídas a partir de padrões e/ou suas representações [60]. Sistemas de reconhecimento de padrões geralmente integram sistemas maiores, nos quais essa atividade é usada para se tomar decisões a partir dos resultados das classificações. A principal vantagem de um reconhecimento automático de padrões está na velocidade com que tarefas de classificação podem ser realizadas, porém estão limitadas ao reconhecimento de classes que foram consideradas na fase de desenvolvimento, podendo usar apenas as características previamente definidas [61].

Há diferentes paradigmas para a solução do problema de reconhecimento de padrões. Destacam-se o *reconhecimento sintático de padrões* e o *reconhecimento estatístico de padrões*. O primeiro paradigma se utiliza das inter-relações de características, usando descrições básicas denominadas primitivas para a descrição da estrutura dos padrões, ao passo que na abordagem estatística as características das classes são regidas por determinados modelos probabilísticos [62].

Um exemplo de reconhecimento de padrões de natureza estatística é o realizado por redes neurais. Essas estruturas realizam o reconhecimento de padrões passando, inicialmente, por uma fase de treinamento, na qual é apresentado repetidamente à rede um conjunto de padrões de entrada juntamente com o rótulo (categoria) à qual cada padrão particular pertence. Posteriormente, são apresentados à rede neural novos padrões, os quais não foram vistos previamente, porém pertencentes à mesma população de padrões empregada para treinar a rede. A rede deve identificar a classe daquele padrão específico devido à informação por ela extraída a partir dos dados de treinamento [63].

### 2.5.4 Reconhecimento de padrões em imagens

O reconhecimento de padrões em imagens tem obtido cada vez mais importância, no contexto de *big data*, para pesquisadores em diversos domínios da ciência e da tecnologia, como medicina e segurança [64]. Esse tipo de reconhecimento, cujo objetivo principal é gerar descrições de imagens e relacionar essas descrições para modelos de modo a categorizar classes de imagens [65], envolve uma série de etapas, as quais são apresentadas na Figura 2.17.

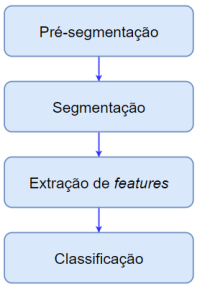



Figura 2.17: Diagrama das fases do reconhecimento de padrões em imagens.

*Pré-segmentação*: fase em que podem ser removidos os ruídos das imagens, eliminando detalhes irrelevantes ou aprimorando *features* (características) pré-selecionadas [66].

*Segmentação*: refere-se ao processo de particionamento (*splitting*) de regiões com propriedades semelhantes de uma imagem. A detecção dessas regiões pode ser realizada por meio de agrupamento de pontos vizinhos com características semelhantes, ou por meio da determinação da borda da região [67].

*Extração de features*: nessa fase objetiva-se buscar alguma informação quantitativa de interesse ou que seja básica para discriminar as classes de objetos, sendo necessário escolher a forma de representação dos dados, podendo ser por fronteiras (externa) ou por regiões completas (interna). Assim, nessa etapa a imagem é representada como um vetor numérico a fim de diminuir a redundância dos dados e efetuar a redução de dimensionalidade [67].

*Classificação*: essa etapa parte da premissa de que a similaridade entre objetos implica que eles possuem características similares, constituindo classes/agrupamentos. De forma geral, essa fase compreende o reconhecimento de um objeto, uma forma, ou seja, de uma entidade particular da imagem [67].

Algoritmos de *data mining* e *machine learning* são frequentemente aplicados no contexto do reconhecimento de padrões em imagens, podendo ser empregados na classificação de imagens ou reconhecimento de elementos específicos. Nesse campo, a maioria dos modelos que constituem o estado da arte são desenvolvidos a partir de *Convolutional Neural Networks* (CNN) [68].

### 2.5.5 Artificial Neural Networks (ANN)

A motivação para o desenvolvimento e o estudo na área de redes neurais artificiais está no reconhecimento de que o cérebro humano é capaz de processar informações de modo diferente de um computador tradicional, sendo esse órgão do corpo humano altamente complexo, não-linear e paralelo. A capacidade de organização dos seus elementos básicos constituintes o permite processar muito mais rapidamente que o mais veloz computador digital existente, como verificado, por exemplo, no sentido da visão. Essa elevada capacidade é devida a sua grande estrutura e a sua habilidade de desenvolver suas próprias regras, por meio do que usualmente é chamado de "experiência" [63].

Uma rede neural artificial, de maneira geral, é uma máquina desenvolvida para modelar a forma como o cérebro desempenha uma tarefa particular ou função de interesse. Essa máquina pode ser desenvolvida utilizando-se componentes eletrônicos ou por meio de simulação por programação em um computador [63].

As redes neurais artificiais (RNA) podem ser definidas como sistemas distribuídos formados por unidades de processamento simples (nós), as quais calculam determinadas funções matemáticas. Essas unidades são dispostas em uma ou mais camadas e estão interligadas por um elevado número de conexões, usualmente unidirecionais. Geralmente essas conexões estão associadas a pesos que armazenam o conhecimento representado no modelo e são utilizados para ponderar a entrada recebida por cada neurônio da rede neural [69].

RNAs possuem a capacidade de aprender por meio de exemplos e de generalizar a informação aprendida, sendo capazes de extrair informações não apresentadas de forma explícita por meio de exemplos [69]. O neurônio artificial, ilustrado na Figura 2.18, unidade básica de uma RNA, é formado basicamente por sinais de entrada, pesos, função de ativação e sinal de saída. Cada entrada $x_i$ da rede neural é multiplicada pelo peso correspondente ($w_i$) e, então, esses valores são somados. Ao resultado dessa soma é adicionado um deslocamento linear conhecido como *bias* ($b$) e, dessa forma, é gerado um sinal de ativação $u$, cujo valor é obtido pelo cálculo $u = \sum_i x_i \times w_i + b$. Esse sinal é passado para uma função de ativação *f(u)* retornando um valor $y$ [70].

Um aspecto importante para a escolha de um determinado tipo de RNA para a solução de um problema específico é a sua arquitetura. Ao se analisar a arquitetura de uma rede neural, parâmetros como número de camadas na rede, número de nós em cada camada, tipo de conexão entre os nós e topologia da rede são elementos importantes para a sua classificação [69].

No que diz respeito ao número de camadas, há redes em que só existe um nó entre as entradas e as saídas (redes de camada única), e redes em que existem mais de um neurônio entre alguma entrada e alguma saída (redes de múltiplas camadas). A Figura

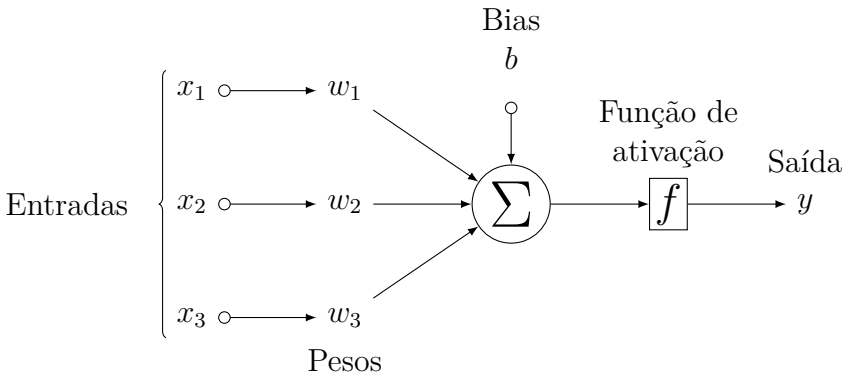



Figura 2.18: Estrutura de um neurônio artificial.

2.19 ilustra, de forma genérica, a estrutura de uma RNA de múltiplas camadas. No que tange ao tipo de conexão, há redes em que a saída de um neurônio na $i$-ésima camada não pode ser usada como entrada de nós em camadas de índice menor ou igual a $i$ (*feedforward* ou acíclica), e redes em que a saída de um neurônio na $i$-ésima camada é empregada na entrada de nodos em camadas de índice menor ou igual a $i$ [69].

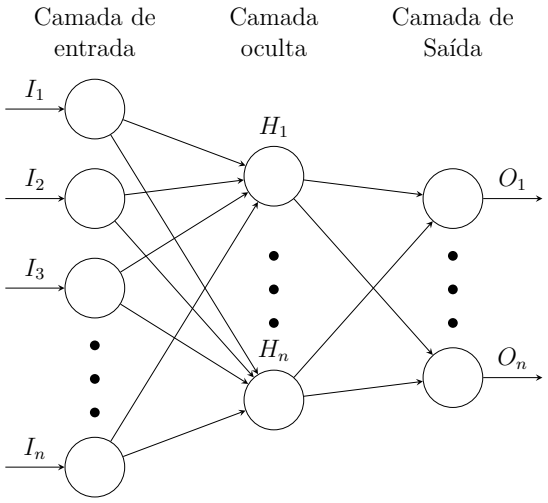



Figura 2.19: Estrutura de uma rede neural artificial com múltiplas camadas.

A função de ativação, também chamada de *função restritiva*, representada por $\varphi(\upsilon)$ presente no neurônio serve para restringir sua saída. Dessa forma, há uma limitação no valor de amplitude do sinal de saída a um valor finito, tipicamente pertencente ao intervalo [0,1] ou [-1,1] [63]. A seguir serão apresentados os principais tipos de função de ativação utilizados em RNA (ver Figura 2.20):

- *Função de limiar*: em neurônios que empregam esse tipo de função sua saída será 1, se o campo local induzido deste neurônio for não-negativo, caso contrário o valor retornado será 0 [63]. Normalmente é utilizado em classificadores binários. A função

liminar pode ser representada matematicamente por:

$$\varphi(v) = \begin{cases} 1 & \text{se } v \geq 0 \\ 0 & \text{se } v < 0 \end{cases}$$

- *Função sigmoide*: tipo de função comumente utilizada apresentando uma estrutura estritamente crescente. Um exemplo de função sigmoide é a função logística, definida matematicamente por:

$$\varphi(v) = \frac{1}{1 + e^{(-av)}}$$

em que a variável $a$ representa o *parâmetro de inclinação* da função sigmoide, sendo possível, portanto, obter funções com diferentes inclinações variando-se o valor de $a$. Diferentemente da função limiar, a sigmoide é diferenciável em qualquer ponto [63].

- *Função tangente hiperbólica*: a função citada no item anterior retorna valores no intervalo de 0 a 1, porém em algumas aplicações, pode ser necessário empregar funções que retornem valores no intervalo de -1 a +1, por exemplo. Nesse tipo de situação, pode-se usar a função tangente hiperbólica, definida matematicamente da forma:

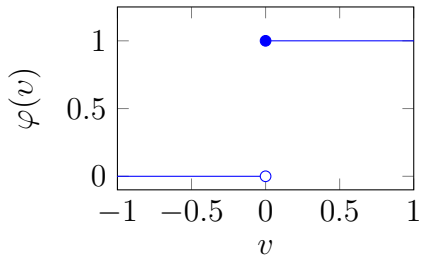
$$\varphi(v) = tanh(v)$$

- *Função ReLU* (*rectified linear unit*): a função de ativação ReLU é a mais utilizada atualmente para o treinamento de redes neurais profundas [71]. Um vantagem dessa função é que ela não ativa todos os neurônios ao mesmo tempo, ou seja, se a entrada for negativa, ela será convertida em 0 e o neurônio não será ativado, assim apenas alguns neurônios serão ativados. A função ReLU pode ser representada da forma

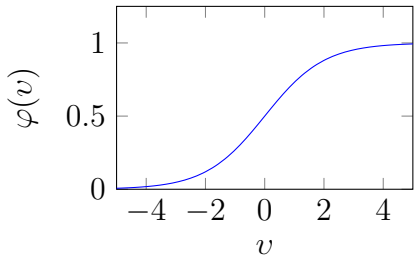
$$\varphi(v) = max(0, v)$$

- *Função softmax* : quando se deseja interpretar a saída como probabilidades, permitindo atribuir um exemplo à classe com maior probabilidade, utiliza-se esse tipo de função. Salienta-se que a função do tipo sigmoide é capaz de lidar com duas classes, ao passo que *softmax* permite trabalhar com um conjunto maior de classes. Essa função recebe um vetor de entradas e fornece um vetor com a probabilidade do pertencimento desse conjunto de entradas para cada classe analisada.

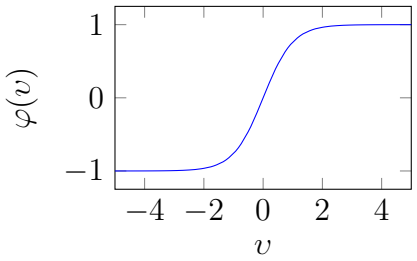
$$softmax(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$$



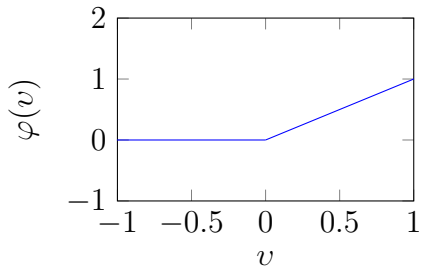

(a) Função de limiar




(b) Função sigmoide




(c) Função tangente hiperbólica




(d) Função ReLU

Figura 2.20: Gráficos das funções de ativação $\varphi(v)$ mais comuns.

Os benefícios gerados por uma RNA surgem devido ao seu poder computacional, o qual é atribuído a sua estrutura maciçamente paralela distribuída e a sua capacidade de aprender e, portanto, generalizar. Essa capacidade de generalização permite o fornecimento de respostas adequadas a entradas novas, não fornecidas durante a etapa de treinamento (aprendizagem) da rede neural. De modo geral, o emprego de redes neurais artificiais possui as seguintes propriedades [63]:

- *Não-linearidade*: Uma rede neural construída por conexões de neurônios não-lineares, também será considerada não-linear. Essa é uma propriedade importante especialmente se o mecanismo físico encarregado por gerar os sinais de entrada for do tipo não-linear.

- *Adaptabilidade*: RNAs possuem a capacidade de adaptar seus pesos sinápticos conforme o ambiente. Uma rede neural treinada pode ser retreinada para refletir as mudanças em um ambiente específico. Para poder valer-se de todos os benefícios da adaptabilidade, um sistema deve possuir suas constantes de tempo grandes o suficientes para não se deixar influenciar por variações espúrias, mas pequenas o suficiente para responder a mudanças significativas no ambiente no qual está inserido.

- *Tolerância a falhas*: como o conhecimento é distribuído ao longo da RNA, a falha em um ponto específico pode não acarretar danos consideráveis em seu desempenho.

- *Resposta a evidências*: no que concerne à classificação de padrões, uma rede neural pode não apenas classificar uma entidade sob questionamento, como também indicar o nível de confiança na decisão tomada. Esse nível de confiança fornecido por RNA permite a melhoria do desempenho de classificação ao rejeitar padrões ambíguos, caso estejam presentes.

### 2.5.6 Convolutional Neural Networks (CNN)

As redes neurais convolucionais (RNC) ou *convolutional neural networks* (CNN) foram desenvolvidas para trabalhar com entradas no formato de *grid*, como imagens bidimensionais. Nesse tipo de entrada verificam-se *dependências espaciais*, em que regiões adjacentes frequentemente apresentam cores similares em seus *pixels*. Apesar de sua grande popularidade por trabalhar com arquivos de imagem, CNN também são empregadas em dados temporais, espaciais e espaço-temporais [72].

As CNN constituem-se em um dos primeiros casos de sucesso de *deep learning*, ganhando bastante visibilidade, principalmente, a partir dos concursos de classificação de imagens realizados a partir de 2011. Esse tipo de arquitetura é adequado para o processo de *feature engineering* do tipo hierárquico [72].

A motivação para o desenvolvimento de redes neurais convolucionais iniciou a partir dos estudos de Hubel e Wiesel [73] acerca do córtex visual dos gatos, o qual é constituído por pequenas regiões de células que são sensíveis a áreas específicas no campo de visão. A excitação de determinadas células depende do formato e das orientações dos objetos vistos. A conexão entre as células ocorre por meio de uma arquitetura em camadas, que possibilita a construção da imagem com diferentes níveis de abstração [72].

As redes neurais convolucionais são um tipo de rede neural artificial em que as camadas iniciais são especializadas em extrair características dos dados (principalmente imagens) e em que os neurônios não são totalmente conectados com os da camada seguinte. As camadas finais, responsáveis por interpretar as informações obtidas por meio das camadas iniciais e gerar uma resposta, são totalmente conectadas [70].

As RNCs são bastante utilizadas no processamento de informações visuais, em particular imagens, devido ao fato de a convolução permitir filtrar as imagens considerando sua estrutura bidimensional (espacial) [74]. De forma similar aos processos tradicionais de visão computacional, uma rede neural convolucional consegue aplicar filtros em dados visuais mantendo a relação de vizinhança entre os *pixels* da imagem durante o processamento da rede [75]. A Figura 2.21 exibe um exemplo de arquitetura de uma RNC.

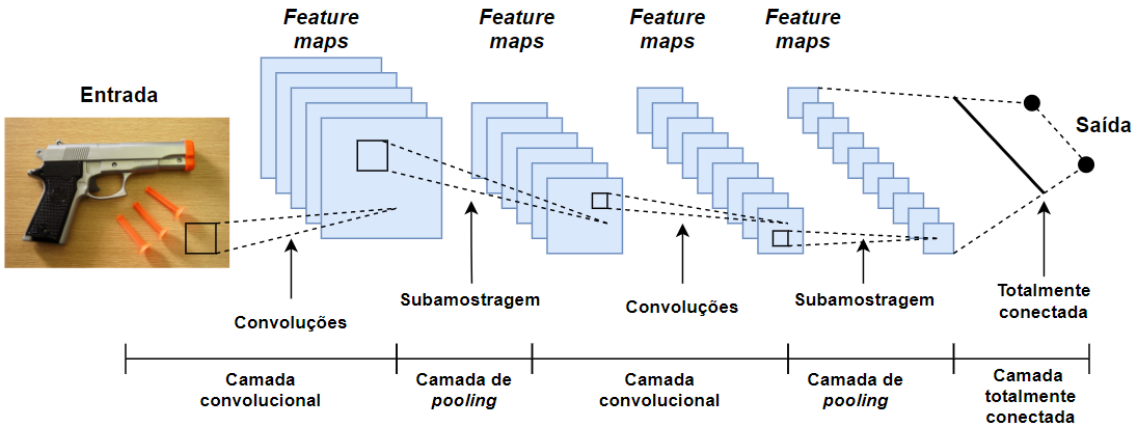



Figura 2.21: Exemplo de arquitetura simples de uma CNN.

No que tange à arquitetura de uma RNC, elas são redes neurais do tipo *feedforward* projetadas de modo a minimizar a sensibilidade à translação, distorção e rotação de uma imagem de entrada. Elas são organizadas em camadas com diferentes funções [70]. Na *camada convolucional*, cada neurônio é um filtro (matriz de pesos) aplicado a uma imagem de entrada. Assim, cada filtro (neurônio) dessa camada processa a imagem produzindo uma transformação dessa entrada por meio de uma combinação linear dos *pixels* vizinhos. Um mapa de características (*feature map* ou *activation map*) constitui-se de cada representação gerada por um filtro da camada convolucional. Esses mapas são empilhados formando um *tensor* de profundidade igual ao número de filtros, o qual será oferecido como entrada para a próxima camada [74].

A fim de diminuir a dimensão espacial dos mapas ao longo das camadas da rede, a operação de *maxpooling* é frequentemente empregada. Essa redução proporcionada pela camada de *pooling* é importante para fins de agilidade no treinamento, porém essa camada não reduz a profundidade da entrada, apenas diminuiu a altura e a largura de um mapa [75]. As camadas *fully connected* são constituídas por neurônios os quais possuem um peso associado a cada elemento do vetor de entrada. Essas camadas são responsáveis por traçar um caminho de decisão a partir de respostas dos filtros das camadas anteriores, para cada classe de resposta. Após as camadas completamente conectadas, a última etapa é a função de classificação, sendo a função *SoftMax* bastante utilizada devido a

sua simplicidade e bons resultados [75]. A seguir serão apresentadas em mais detalhes as camadas que constituem uma RNC.

### Camadas convolucionais (*convolutional layers*)

Este é o principal tipo de camada de uma rede neural convolucional. Ela contém uma série de filtros (também chamados de *kernels*) que realizam a etapa de *convolução* em uma dada entrada e geram um conjunto de *feature maps* ou *activation maps* [76].

Um *filtro* de uma camada de convolução é um *grid* de números discretos (pesos), os quais são aprendidos durante a etapa de treinamento da CNN. Inicialmente os pesos dos filtros são escolhidos de forma aleatória, ao passo que durante o treinamento esses valores são modificados [76]. Na operação de *convolução*, o filtro é colocado em cada posição da imagem ou da camada oculta, de modo que haja a sobreposição da imagem pelo *kernel*. Em seguida é realizada a operação de produto escalar (*dot product*) entre os parâmetros do filtro e da região da imagem sob esse filtro. Essa etapa gerará um conjunto de *feature maps* cuja quantidade dependerá do número de filtros da camada anterior. A Figura 2.22 exibe um exemplo de operação de convolução utilizando um filtro, e o seu correspondente resultado na *feature map*.

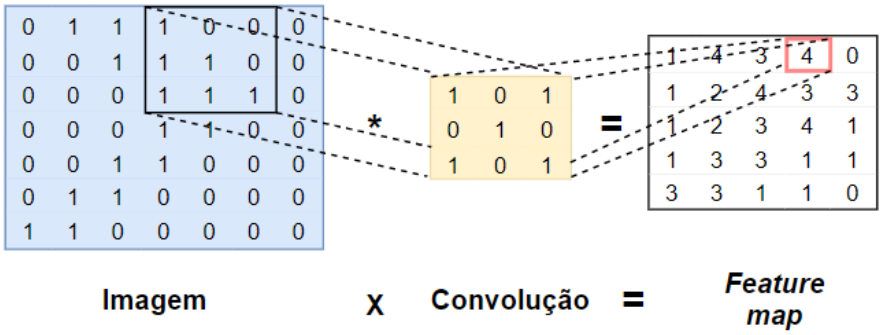



Figura 2.22: Exemplo de uma operação de convolução.

A operação de convolução produz uma redução na camada *q+1*, se comparado com o tamanho da camada *q* anterior. Essa redução no tamanho pode não ser desejada, uma vez que informações presentes na borda da imagem, ou no *feature map* no caso das camadas ocultadas, são perdidas. Esse problema pode ser resolvido com a inclusão do *padding*, em que *pixels* extras são adicionados na borda do elemento a ser aplicada a operação de convolução [77]. Um elemento que pode reduzir o nível de granularidade de uma convolução é o *stride* [77]. Com o objetivo de calcular cada valor de saída no *feature map*, um filtro pode se deslocar em "passos" (*strides*) de uma ou mais unidades [76].

### Camada de *pooling* (*Pooling layer*)

A operação de *pooling* é realizada em *grids* de tamanho $P_q$ x $P_q$ em cada camada, e produz uma camada de mesma profundidade, diferentemente dos filtros. Para cada região quadrada de dimensões $P_q$ x $P_q$ de cada *activation map* $d_q$ é retornado o valor máximo desse *grid*. A esse tipo de *pooling* dá-se o nome de *max pooling*. Assim, por meio dessa operação é possível reduzir significativamente as dimensões espaciais de cada *activation map* [77].

É comum a utilização da operação de *pooling* com filtros 2x2 e *stride* 2, quando é desejado reduzir o espaço das *activation layer* [77], conforme apresentado na Figura 2.23. Essa redução na dimensão das *feature maps* é útil para a obtenção de uma representação compacta que seja invariante a mudanças moderadas de escala, posição e translação em uma imagem [76].

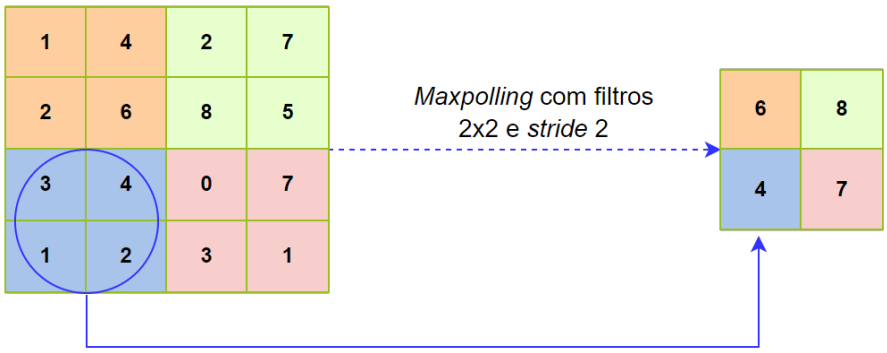



Figura 2.23: Exemplo de operação *max pooling* em um *activation map* de tamanho 4x4.

### Camada ReLU

A operação de convolução é intercalada com as operações de *pooling* e ReLU. A função de ativação ReLU é aplicada para cada $L_q$ x $B_q$ x $dq$ valores de uma camada a fim de limita-los, passando os valores resultantes para a próxima camada. Assim, como trata-se apenas de um mapeamento *um para um* dos valores de ativação, não há mudanças nas dimensões da camada [77].

A função ReLU realiza o mapeamento da entrada para 0, se seu valor for negativo, ou mantém seu valor inalterado, caso seu valor seja positivo [76], conforme apresentado na Seção 2.4.5. Essa função de ativação representa uma evolução no *design* de redes neurais artificiais, substituindo funções comumente empregadas, como sigmoide e tangente hiperbólica. Essa mudança ocorre devido ao aumento na velocidade e na acurácia propiciado pela função de ativação ReLU [77].

#### Camada totalmente conectada (*fully connected layer*)

As camadas totalmente conectadas correspondem a camadas convolucionais com filtros de tamanho 1x1. Cada unidade de uma camada totalmente conectada é densamente conectada a todas as unidades da camada anterior. Tipicamente esse tipo de camada é colocado no final de uma CNN, porém há trabalhos que empregam a *fully connected layer* em outras posições na arquitetura [76].

### 2.5.7 Exemplos de arquiteturas CNN

A seguir serão apresentados exemplos de arquiteturas de *Convolutional Neural Networks*. Essas arquiteturas serão empregadas na etapa de transferência de aprendizado *transfer learning* do projeto, presente na Seção 3.4.

#### a) VGG

A arquitetura VGG (*Visual Geometry Group*) foi proposta por Simonyan e Zisserman [78], que investigaram o efeito da profundidade de uma rede convolucional na acurácia na classificação de imagens em larga escala. A arquitetura VGG não foi a vencedora do desafio ILSVRC (*ImageNet Large Scale Visual Recognition Challenge*) de 2014, que envolvia a classificação aproximada de 14 milhões de imagens em cerca de 1000 categorias, mas esteve entre as primeiras colocadas e introduziu novidades importantes na área.

Uma relevante inovação trazida pela VGG é a redução no tamanho dos filtros, aumentando a profundidade da rede, se comparada com arquiteturas anteriores. Outro elemento importante implantando nesse modelo foi o aumento na quantidade de filtros na potência de 2 depois de cada camada *max pooling*. Essa inovação no design da arquitetura foi empregada em outros modelos, como ResNet [72].

A arquitetura VGG com melhor desempenho no ILSVRC 2014 apresentava 16 camadas, sendo constituídas por *convolutional layers, max pooling layers, activation layers* e *fully connected layers*. Ao todo há 13 camadas de convolução, 5 camadas de *max pooling* e 3 camadas densas. Durante o treinamento a rede foi treinada com imagens RGB de tamanho fixo de 224x224 [78]. A imagem era passada para uma pilha de camadas convolucionais, onde filtros de tamanho 3x3 eram utilizados. Uma unidade linear de retificação (ReLu) é executada após cada convolução e uma operação de *max pooling* é realizada no final de cada bloco para redução da dimensão espacial [79]. Enquanto que a arquitetura VGG16 possui 13 camadas convolucionais e 3 totalmente conectadas, a VGG19 possui 16 camadas convolucionais e 3 totalmente conectadas. As arquiteturas VGG16 e VGG19 estão esquematizadas na Figura 2.24 [80].

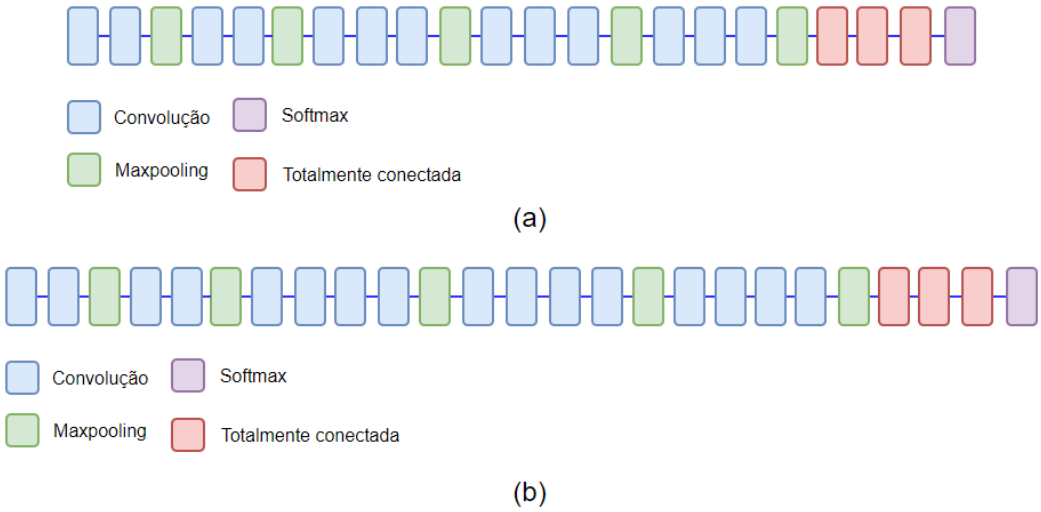



Figura 2.24: Diagrama das estruturas dos modelos (a) VGG16 e (b) VGG19.

Esses modelos apresentam como pontos negativos a baixa velocidade durante a fase de treinamento e a grande quantidade de parâmetros utilizada (cerca de 128 milhões), impactando o consumo de memória. Uma das maiores contribuições do projeto VGG foi demonstrar que a profundidade de uma rede é um componente crítico para a obtenção de melhores resultados no reconhecimento ou classificação em redes neurais convolucionais [81].

### b) GoogLeNet

A arquitetura VGG apresentada consistia de uma rede sequencial composta por um caminho simples, ou seja, sua estrutura era linear e formada por diferentes tipos de camadas, como as convolucionais, *pooling*, ReLu e totalmente conectadas [76]. A arquitetura GooLeNet inovou ao introduzir o conceito de *inception arquitecture*, que pode ser visto como uma "rede dentro de uma rede" [72]. Salienta-se que essa arquitetura conquistou a primeira colocação no ILSVRC 2014, deixando a VGG em segundo lugar.

O modelo consistia de 22 camadas, sendo construído com blocos básicos chamado *inception modules*, cuja estrutura está apresentada na Figura 2.25, daí essa arquitetura ser chamada de *inception network*. Diferentemente do processamento sequencial da VGG, o processamento no módulo ocorria de forma paralela [76]. A ideia central da utilização do módulo *inception* é que informações chaves das imagens são disponíveis em diferentes níveis de detalhes. Com a utilização de filtros maiores é possível capturar informações em uma área maior contendo variação limitada, ao passo que com filtros menores torna-se possível capturar informações mais detalhadas de uma área menor. Assim, ao utilizar

filtros de tamanhos diferentes em suas convoluções, o módulo *inception* permite maior flexibilidade ao modelo, fornecendo diferentes níveis de granularidade [72].

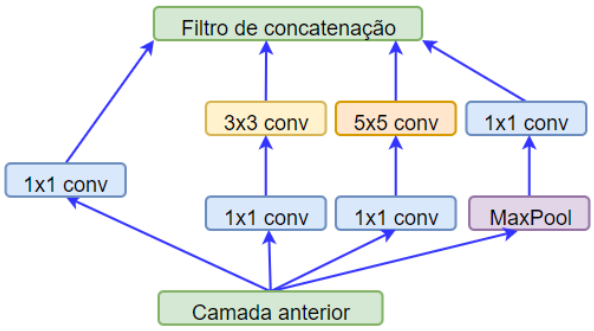



Figura 2.25: Estruturas do módulo *inception*.

Apesar de sua maior complexidade que as arquiteturas anteriores, GoogLeNet apresenta menor número de parâmetros (aproximadamente 6 milhões, contra os 128 milhões presentes na VGGnet), sendo bastante eficiente e possuindo elevada acurácia [76]. Essa redução no número de parâmetros deve-se, principalmente, pelo uso da *average pooling*, que se tornou padrão em muitas arquiteturas posteriores [72]. A estrutura da arquitetura GooLeNet está exposta na Figura 2.26.

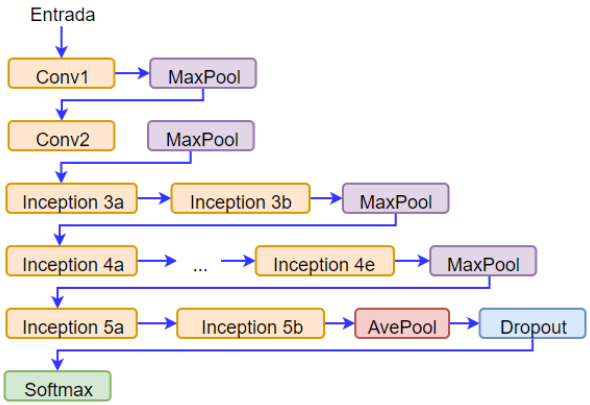



Figura 2.26: Arquitetura GoogLeNet.

### c) Xception

A arquitetura Xception, proposta por Chollet [82], é semelhante ao GoogLeNet se diferenciando por introduzir o conceito de *depthwise separable convolutions*, em substituição à unidade *inception*. A arquitetura Xception é constituída por uma pilha de camadas convolucionais do tipo *deepwise separable* com conexões residuais. Inicialmente a arquitetura aplica uma convolução 1x1 *pointwise* e, logo depois, aplica uma convolução de 3x3

em profundidade, conforme apresentado na Figura 2.27. Xception é composta por 36 camadas de convolução estruturada em 14 módulos.

Mesmo apresentando o mesmo número de parâmetros da arquitetura InceptionV3, Xception apresentou melhores resultados na classificação do banco de imagens do ImageNet. Isso evidencia que esse incremento na performance ocorreu não devido ao aumento da capacidade, mas sim ao uso mais eficiente do modelo de parâmetros [82].

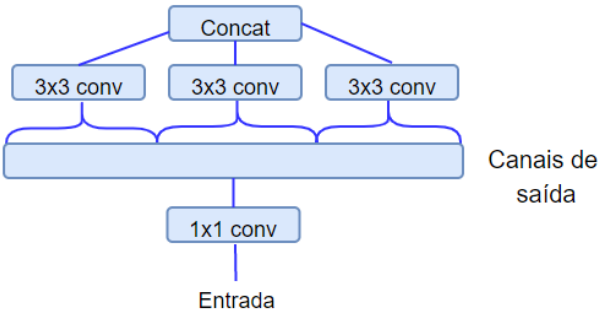



Figura 2.27: Estrutura da arquitetura Xception.

### d) ResNet

A arquitetura ResNet obteve a maior acurácia do ILSVRC 2015, apresentando uma estrutura formada por mais camadas, comparando-se com as arquiteturas anteriores. Dois possíveis problemas que podem ser encontrados ao treinar redes profundas com grande quantidade de camadas são o *vanish gradient problem* (problema da dissipação do gradiente) e *exploding gradient problem* (problema da explosão do gradiente) devido ao impedimento ao fluxo do gradiente entre as camadas em virtude do grande número de operações [72]. Assim, ResNet deve apresentar uma estrutura que considere esses possíveis problemas.

ResNet é de uma rede neural residual [83] que aplica o conceito de bloco residual, empregando caminhos diretos (*skip connections*) entre suas camadas. Esse atalho copia a entrada de uma camada $i$ e a adiciona à saída de outra camada, facilitando o fluxo do algoritmo de *backpropagation*.

A arquitetura ResNet com melhor desempenho no ILSVRC 2015 é constituída de 152 camadas, formadas pelos blocos básicos residuais. Inicialmente a arquitetura ResNet previa a conexão apenas entre a camada $i$ e outra camada $i+r$ (sendo $r$ um número natural), ao passo que a arquitetura DenseNet já previa a conexão entre todos os pares de camadas, apresentando performance superior. A unidade básica da ResNet pode ser visualizada na Figura 2.28.

A utilização de *skip connections* permite a criação de caminhos livres para o fluxo do gradiente influenciando no comportamento do algoritmo de *backpropagation*, ao interligar

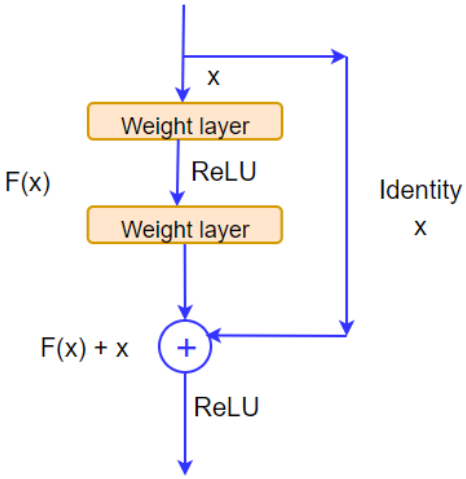



Figura 2.28: *Skip connections* em um bloco residual.

entrada e saída. Esse cenário fornece flexibilidade ao algoritmo de aprendizado ao possibilitar o adequado nível de não linearidade para uma determinada entrada. Entradas mais complexas, por exemplo, podem percorrer um maior número de conexões a fim de se extrair suas *features* relevantes. Essa abordagem é chamada *residual learning* (aprendizado residual), em que o aprendizado dado pelas vias mais longas da rede pode ser considerado uma espécie de *fine-tunning* do aprendizado ocorrido pelos menores caminhos [72].

Dessa forma, a adoção de blocos residuais demonstrou ser um elemento importante para o aumento da acurácia na tarefa de classificação ao se utilizar redes mais profundas. Arquiteturas posteriores foram baseadas em conceitos utilizados em ResNet, como a arquitetura ResNeXt, a qual combina os conceitos de *inception* (ramificações) e *skip connections* (conexões diretas entre determinadas camadas) verificada em ResNet [76].

### 2.5.8 Transferência de aprendizado (*transfer learning*)

Para o correto funcionamento de muitos algoritmos de *machine learning* deve-se ter conjuntos de treinamento e teste representativos de um mesmo espaço e possuindo iguais distribuições. Quando há uma mudança na distribuição de frequência, grande parte dos modelos estatísticos devem ser refeitos utilizando os novos dados coletados para o conjunto de treinamento, o que pode ser caro ou até mesmo inviável em aplicações reais. Diante dessa situação, em que deve-se reduzir a necessidade e o esforço para o reobtenção de um conjunto de treinamento, *transfer learning*, ou transferência de aprendizado, pode ser desejável. Assim, *transfer learning* permite que domínios, tarefas e distribuições utilizadas nos conjuntos de treinamento e teste sejam diferentes [84].

Ressalta-se que o ser humano é capaz de aplicar o conhecimento adquirido na solução de novos problemas de forma mais rápida e com melhores resultados [84]. *Transfer learning* pode ser definido como uma técnica em que um modelo treinado para resolver um problema específico é reutilizado para a solução de um novo problema relacionado ao primeiro. Por exemplo, no reconhecimento de imagens, pode-se ter as primeiras camadas de uma rede responsáveis por detectar características genéricas, ao passo que as últimas camadas sejam capazes de detectar *features* mais específicas [85].

Comparando-se a transferência de aprendizado com as técnicas tradicionais de *machine learning*, verifica-se que as abordagens tradicionais tentam aprender cada tarefa do zero, enquanto que *transfer learning* permite a transferência do conhecimento adquirido a partir de atividades pretéritas para uma nova tarefa alvo. A comparação entre essas abordagens é ilustrada na Figura 2.29 [84].

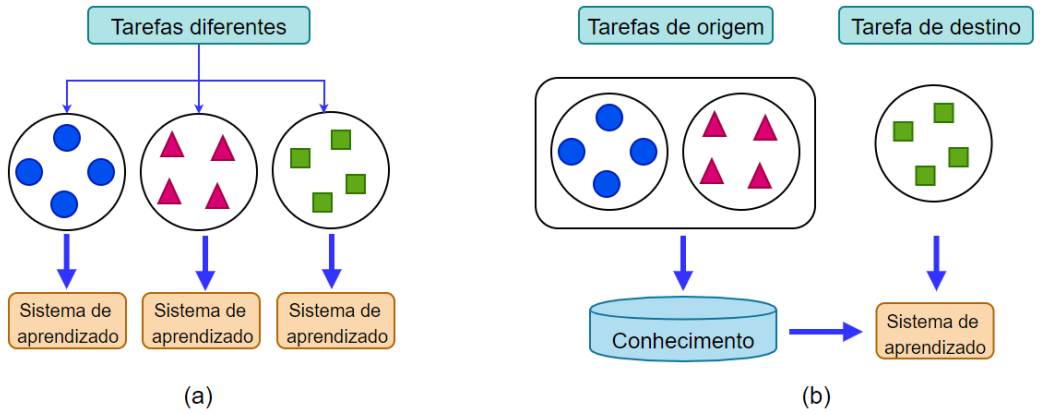



Figura 2.29: Diferença entre os processos de aprendizado entre (a) tradicional *machine learning* e (b) *transfer learning*.

Ao se considerar a possibilidade de se utilizar *transfer learning*, três questões devem ser analisadas: o que transferir (que parte do conhecimento deve ser transferida para outras domínios ou tarefas), como transferir (que algoritmos devem ser empregados para transferir esse conhecimento) e quando transferir (em que situações o conhecimento deve ou não ser transferido) [84]. Há necessidade de uso da transferência de aprendizado quando há uma quantidade limitada de elementos para o conjunto de treinamento. Essa limitação pode ser verificada devido à raridade do dado, ao custo elevado para se obtê-lo e categoriza-lo ou devido a sua inacessibilidade [86].

Métodos de *transfer learning* podem ser categorizados de acordo com o tipo de algoritmo de *machine learning* tradicional envolvido. Assim, tem-se a *inductive transfer*, quando os domínios de origem e de destino são os mesmos sendo as tarefas inicial e final diferentes, o *unsupervised transfer*, quando dados categorizados estão indisponíveis nos dois domínios, e *transductive transfer*, quando há similaridades entre as tarefas de origem

e destino, porém os domínios correspondentes são diferentes. Modelos de *deep learning* representam a aprendizagem indutiva, cujo objetivo é inferir um mapeamento a partir de um conjunto de exemplos de treinamento. Por exemplo, no caso do problema de classificação, o modelo aprende a mapear características de entrada em determinadas classes. A fim de generalizar essa tarefa de classificação para novos dados, o algoritmo assume que esses novos elementos possuem distribuição relacionada aos do conjunto no qual foi treinado [87].

Um pré-requisito para *transfer learning* é a existência de modelos com bons resultados em uma determinada tarefa. Modelos pré-treinados como VGG, AlexNet e Inception, podem ser utilizados em aplicações que envolvam transferência de aprendizado e visão computacional [87]. Em problemas de classificação na área de visão computacional, com uma arquitetura CNN, as primeiras camadas podem desempenhar atividades mais genéricas no processo de classificação, como detecção de bordas, enquanto que camadas finais detectariam elementos mais específicos e relacionados ao domínio alvo que se deseja classificar, com o emprego da transferência de aprendizado.

### 2.5.9 Métricas de avaliação

O modelo de referência CRISP-DM apresenta, em uma de suas fases, a necessidade de se realizar avaliações no modelo. A avaliação da performance de um algoritmo de aprendizagem de máquina pode ser obtida por meio de diversas métricas, as quais exprimem o quão próximas estão as predições realizadas pelo modelo do valor ou classificação esperados. Para modelos de classificação, métricas como acurácia, sensitividade, especificidade, curva ROC (*Receiver Operating Characteristics*) e AUC (*Area Under ROC Curve*) são comumente utilizadas.

Uma *matriz de confusão, matriz de contingência* ou *matriz de erro*, é uma tabela que apresenta a relação entre os valores observados e os valores preditos em um problema de classificação, servindo de base para o cálculo de várias métricas de acurácia [88]. Para um problema de classificação binária, os quatro indicadores presentes em uma matriz de confusão são: *verdadeiros positivos* (VP) (item da classe positiva classificação como positivo), *verdadeiro negativo* (VN) (item de classe negativa classificado como negativa), *falso positivo* (FP) (item da classe negativa classificada como positivo) e *falso negativo* (FN) (item de classe positiva classificado como negativo) [89]. A Figura 2.30 apresenta a matriz de confusão para um problema de classificação binária.

A *acurácia*, derivada da matriz de confusão, exprime a concordância entre as classes preditas e as observadas, representando a proporção entre as predições corretas. Um classificador com boa acurácia deve ter a maioria das tuplas representadas na diagonal principal da matriz de confusão e deve ter valores próximos de zero nas demais células

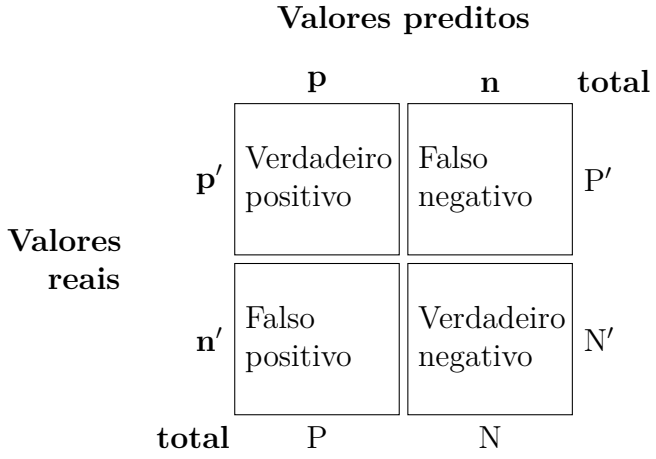



Figura 2.30: Matriz de confusão.

[90]. Seu valor é dado pela soma dos valores preditos corretamente (diagonal principal) e a soma de todo o conjunto de dados, sendo definida por [91]

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN}$$

A *precisão* apresenta a proporção de verdadeiros positivos em relação a todas as predições positivas, ou seja, a taxa com que todos os exemplos classificados como positivos são, de fato, positivos. É expressa da forma

$$Precisão = \frac{VP}{VP + FP}$$

A *sensibilidade* (revocação ou *recall*) é uma métrica calculada com base na proporção de verdadeiros positivos, ou seja, a capacidade do sistema em predizer corretamente a condição para casos que realmente a tem, sendo denotada pela expressão [91]

$$Sensibilidade = \frac{VP}{VP + FN}$$

A partir dos cálculos das métricas anteriores, é possível obter o indicador *F1 score* (*F-measure* ou *F-score*). Esse indicador é obtido realizando-se a média harmônica entre *precisão* e *recall* conforme exposto na expressão

$$F1\ score = 2 * \frac{precisão * recall}{precisão + recall}$$

No que diz respeito à avaliação de modelos de detecção de objetos, a intersecção sobre união ($IoU$) é a métrica mais popular [92]. Ela é empregada para comparar a similaridade de duas formas quaisquer $A, B \subseteq \mathbb{S} \in \mathbb{R}^n$, sendo representada matematicamente na forma da Equação 2.2 e graficamente pelo esquema presente na Figura 2.31.

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{2.1}$$

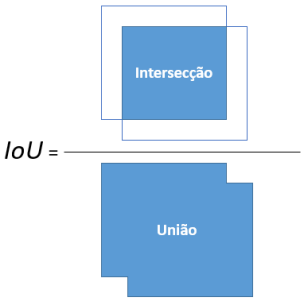



Figura 2.31: Representação gráfica da métrica $IoU$.

A Figura 2.32 ilustra exemplos de taxas $IoU$ para diferentes detecções do objeto arma. Considerando-se o retângulo de contorno verde a marcação real do objeto e o retângulo de contorno amarelo a detecção prevista pelo modelo, verifica-se um aumento no valor de $IOU$ das imagens (a) a (c) presentes na figura. Assim, conforme a detecção do objeto de interesse torna-se mais precisa o valor de $IoU$ aproxima-se de 1.

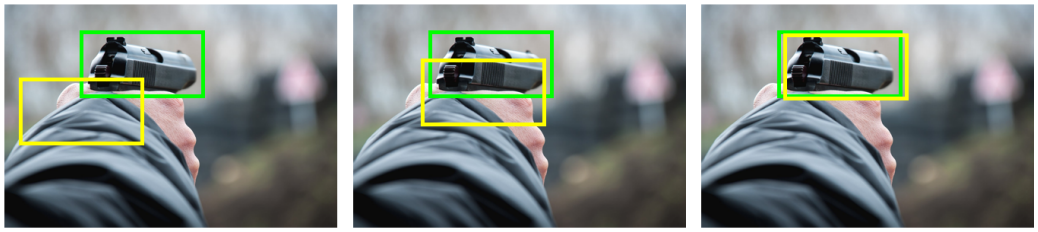



Figura 2.32: Exemplos de detecções do objeto arma. Observa-se um incremento da taxa $IoU$ conforme ocorre maior sobreposição das áreas em destaque.

Outra métrica comumente empregada em tarefas de detecção de objetos é a *mean average precision* (mAP). Para o cálculo dessa métrica são utilizados os conceitos de *precisão* e *recall* definidos nesta Seção. A partir desses dois valores, um gráfico denominado curva PR pode ser expresso, em que o eixo y corresponde à métrica *precisão* e o eixo y, à métrica *recall*. A área presente sob esta curva é chamada de *Average Precision*, enquanto que o o valor de mAP é dado pela média de *Average Precision* calculado para todas as classes de objetos sob detecção.

### 2.5.10 Detecção e localização de objetos

As redes neurais convolucionais podem ser aplicadas em diversas atividades como as relacionadas à visão computacional e a processamento de textos. No que se refere à aplicabilidade em arquivos de imagem e vídeo, por exemplo, destacam-se as atividades de localização de objetos e de detecção de objetos. Na *localização de objetos* há um conjunto

de objetos em uma imagem e deseja-se identificar, por meio de regiões retangulares, cada um desses elementos. Enquanto isso, a *detecção de objetos* distingue-se da abordagem anterior por exibir o nome da classe a que pertence cada objeto presente na imagem [72].

Uma abordagem simples para a localização de objetos consiste na utilização de uma janela deslizante (*sliding window approach*). Nessa abordagem, a imagem é percorrida por uma por uma janela e seu conteúdo é fornecido a um algoritmo de classificação para identificar a presença de um objeto. Essa abordagem trata a detecção de objetos como uma classificação. Para executar uma busca em múltipla escala, a imagem de entrada é rearranjada na forma de uma pirâmide, que a janela percorrerá cada nível, armazenando os resultados do classificador [93].

A abordagem da janela deslizante para a detecção de objetos apresenta como principal limitação o alto custo computacional para varrer a imagem considerando suas escalas e posições. Ao considerar a detecção de cada classe como uma classificação binária, há impacto no tempo de classificação, uma vez que ela será proporcional ao número de categorias analisadas. Além disso, ao se analisar cada janela de forma independente, não se está considerando o contexto no qual ela está inserida [93].

Em virtude dessas limitações, outras abordagens para a detecção de objetos foram estudadas, como a abordagem baseada em regiões. A ideia central desse método é servir como um detector de objeto genérico que une regiões com valores de *pixels* similares, a fim de criar regiões maiores. Inicialmente é criado um conjunto de regiões candidatas e, então, é aplicado o algoritmo de classificação em cada uma delas [77].

Os modelos de detecção de imagens podem ser classificados em duas categorias: *two-stage models* e *one-stage models*. A Figura 2.33 apresenta exemplos de algoritmos pertencentes a cada categoria.

- *Two-stage models*: essa abordagem é inspirada na tarefa de classificação de imagens. Na maioria desses modelos, o primeiro estágio é reservado à extração de regiões (*region proposals*) seguida da etapa de classificação de imagens sobre a referida região. No primeiro estágio (*region proposal*), técnicas como janela deslizante (*sliding window*) e *selective search* podem ser empregadas. Essa abordagem possui elevada acurácia, sendo capaz de identificar objetos em diferentes escalas. Porém, quando se trata de detecção em tempo real, modelos em dois estágios apresentam limitações em virtude da sua velocidade de processamento. Os algoritmos R-CNN, Fast R-CNN e Faster R-CNN são exemplos de modelos *two-stage* [94].

- *One-stage models*: com os avanços no poder computacional foi possível desenvolver CNNs cada vez mais profundas capazes de detectar objetos de forma mais célere e acurada. Modelos do tipo *one-stage* geralmente dividem a imagem em *grids* de

tamanho NxN, sendo cada célula de um *grid* responsável pelo objeto cujo centro esteja situado em seu domínio. Modelos *one-stage* apresentam uma arquitetura mais simples, exigindo, geralmente, menos recursos computacionais. As arquiteturas YOLO (*You Only Look Once*) e SSD (*Single-Shot Multibox Detector*) são exemplos dessa categoria de modelos [94].

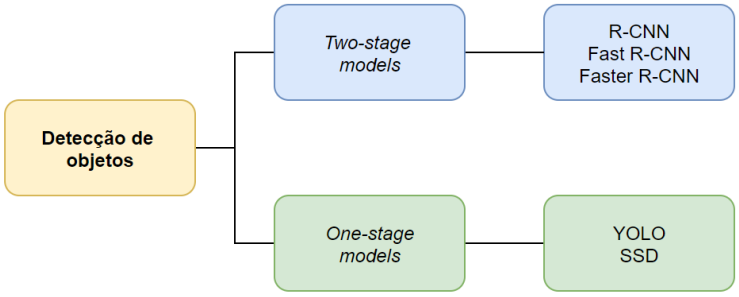



Figura 2.33: Categorias de modelos para detecção de objetos e exemplos de algoritmos.

### *Region-based* CNN (R-CNN)

Essa estratégia, proposta por Girshick et al. [95], utiliza o método de *region proposal* descrito nesta Seção. Esse método de detecção de objetos emprega três módulos, conforme apresentado na Figura 2.34.

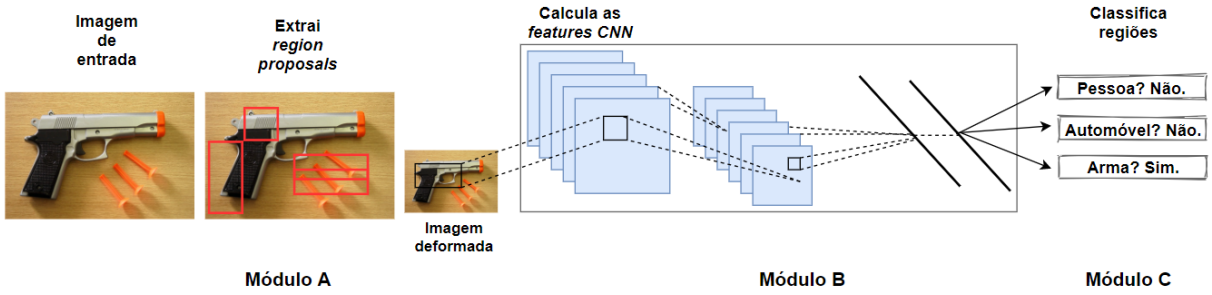



Figura 2.34: Estrutura do sistema de detecção de objetos R-CNN.

- *Regional proposals* (Módulo A): dada uma imagem de entrada é utilizado o algoritmo *selective search* a fim de obter um conjunto de regiões candidatas a terem possíveis objetos.
- *Feature extraction* (Módulo B): essa etapa utiliza uma CNN treinada, como AlexNet ou VGGnet, para extrair um vetor de *features* de tamanho fixo para cada região. Para essa etapa, a região é convertida a fim de manter a compatibilidade com a entrada da rede utilizada [76].

- *Classificador* (Módulo C): após a extração de *features* um SVM linear por classe é utilizado consistindo no terceiro módulo do sistema de detecção [76].

O método de detecção R-CNN apresenta relevantes desvantagens, como o elevado custo de treinamento. O procedimento é lento, uma vez que a CNN deve ser executada para cada região de interesse (*region of interest - RoI*) separadamente. Assim, essa abordagem resultará em grande consumo computacional nos casos em que houver sobreposição de RoI [96].

### Fast R-CNN

O algoritmo Fast R-CNN proposto por Girshick [97] introduz o conceito de *RoI pooling layers*, que visa projetar as *features* fornecidas pela última camada convolucional em um vetor de *features* de tamanho fixo. Assim como o algoritmo R-CNN, Fast R-CNN também possui como entrada uma imagem e RoIs, porém é obtido uma *feature map* da imagem por meio da rede convolucional [96]. A Figura 2.35 apresenta a arquitetura Fast R-CNN.

A camada de *RoI pooling* usa *max pooling* para converter o *feature map* de cada RoI em um vetor de tamanho fixo. O algoritmo retornará a classe do objeto com maior probabilidade, bem como o *bounding box* para cada objeto. O algoritmo Fast R-CNN utiliza o modelo VGG16, apresentando velocidade e acurácia superiores às verificadas em R-CNN. Porém, da mesma forma que o algoritmo R-CNN, essa abordagem também apresenta como ponto negativo a necessidade de obtenção das regiões de interesse (RoI) por um algoritmo externo [96] e não pela própria rede neural convolucional.

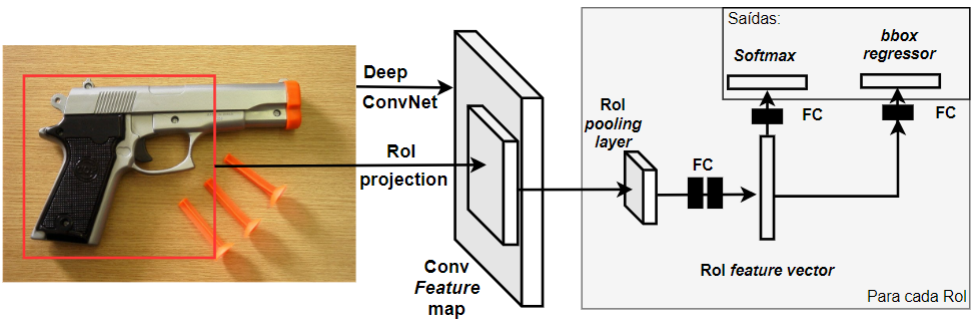



Figura 2.35: Arquitetura Fast R-CNN.

### Faster R-CNN

A qualidade da detecção de objetos do algoritmo Fast R-CNN é dependente da qualidade do processamento das *region proposals*. Assim, para contornar essa limitação, Ren et al. [98] propuseram o algoritmo Faster R-CNN, o qual insere a unidade *Region Proposal Network* (RPN) após a última camada convolucional. O objetivo dessa rede RPN é

determinar regiões de interesse (RoIs) que serão fornecidas como entrada ao algoritmo Fast R-CNN [96]. A arquitetura Faster R-CNN está ilustrada na Figura 2.36.

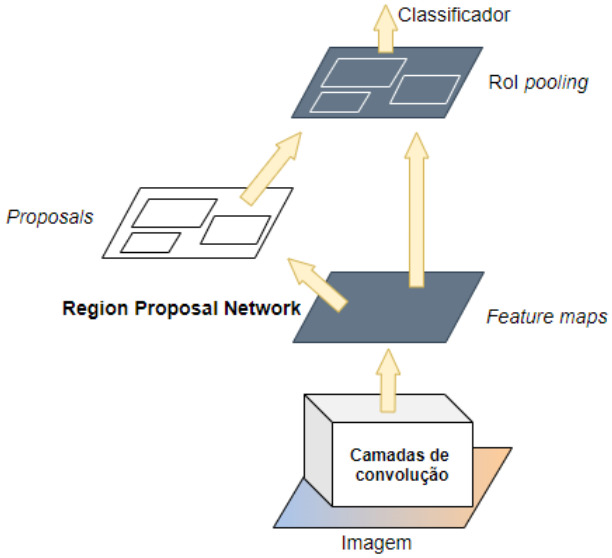



Figura 2.36: Arquitetura Faster R-CNN.

A arquitetura Faster R-CNN é constituída por dois módulos: uma rede convolucional profunda que realiza o processamento de regiões, e o detector de objetos Fast R-CNN, que utiliza as regiões propostas. Em virtude dessa computação conjunta com o modelo Fast R-CNN, as duas redes devem compartilhar um conjunto comum de camadas convolucionais [98].

A estrutura RPN presente na arquitetura atua na forma de uma janela deslizante (*sliding window*) sobre o *feature map* graças a uma camada convolucional com filtros de tamanho pequeno, como 3x3. Assim, para cada posição da janela deslizante, outra *feature* é calculada, a qual é fornecida como entrada para duas camadas totalmente conectadas (*fully-connected layers*). É função dessas camadas determinar se há um objeto presente na região, bem como fornecer o *bounding box* nos casos positivos para a presença de objetos [96].

### YOLO (*You Only Look Once*)

Diferentemente das abordagens anteriores, o algoritmo proposto por Redmon [99] considera a detecção de objetos como um problema de regressão. Ao invés de gerar *region proposals*, o algoritmo YOLO realiza a predição direta de múltiplas *bounding boxes* a partir da imagem de entrada [20]. O modelo é treinado a fim de identificar *bounding boxes*, juntamente com suas probabilidades de conterem um objeto. Desse modo, ao invés de gerar explicitamente regiões de interesse (RoIs) como janelas candidatas a conterem pos-

síveis objetos, em que para cada *proposal* é realizada uma detecção, YOLO processa cada imagem uma única vez realizando múltiplas detecções. Assim, esse método de detecção apresenta avanços em termos de velocidade [100].

Além disso, ao processar uma imagem inteira e não apenas regiões específicas, o algoritmo é capaz de extrair informações contextuais de forma mais apurada. Assim, verifica-se uma menor taxa de detecção do fundo (*background*) como objeto, impactando positivamente a quantidade de falsos positivos comparando-se com outros métodos [100].

No que tange ao processo de detecção de objetos YOLO, a imagem de entrada é redimensionada para 448x448 *pixels*, sendo executada então uma rede neural convolucional nessa imagem, obtendo-se *bounding boxes* e seus respectivos *scores* de confiança. Após essa etapa, um módulo NMS (*no-max suppression*) é utilizado para limitar os resultados das detecções, resultando nas probabilidades de cada classe associadas às coordenadas de seus *bounding boxes* [101]. O processo de detecção de objetos YOLO está apresentado na Figura 2.37.

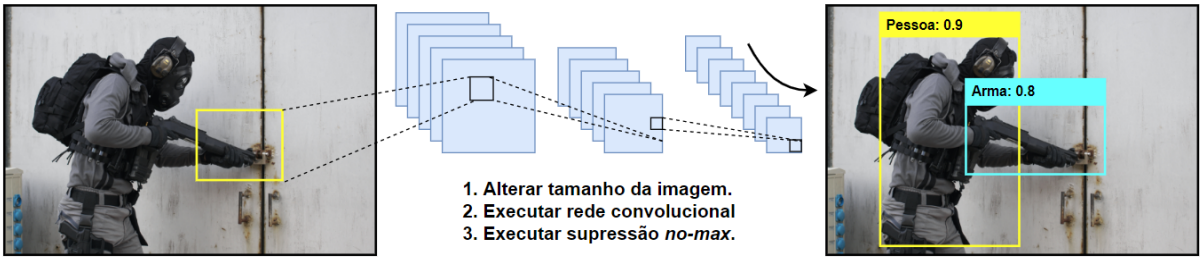



Figura 2.37: Processo de detecção de objetos do algoritmo YOLO.

O algoritmo YOLO inicialmente divide a imagem em *grids* de tamanho SxS. Se há o centro de um objeto no interior de um determinado *grid*, essa célula será a responsável pela detecção do objeto. Cada célula deve fazer a predição de $B$ *bounding boxes*, sendo cada uma dessas áreas retangulares descritas por cinco parâmetros, $x$, $y$, $w$, $h$ e *confiança (confidence)*, em que o par ordenado $(x, y)$ representa o centro do retângulo referente aos limites da célula, e $(w, h)$ representam, respectivamente, a largura e a altura relativas às dimensões da imagem de entrada [94]. O *score* de confiança pode ser definido matematicamente como $Pr(Object) * IOU^{real}_{previsto}$. Se não há objetos na célula, o *score* de confiança deverá ser zero, caso contrário será igual ao valor da *intersection over union* (IOU), denotada pela divisão da área de sobreposição pela área da união [99].

A arquitetura YOLO, exposta na Figura 2.38, é inspirada no modelo de classificação de imagens GoogLeNet, contendo 24 camadas convolucionais seguidas de 2 camadas totalmente conectadas. Em vez dos módulos *inception* presentes na arquitetura GoogLeNet, YOLO utiliza camadas de redução de 1x1 seguidas por camadas de convolução de 3x3. As camadas convolucionais são pré-treinadas utilizando-se as imagens do ImageNet,

composta de 1000 categorias diferentes. O modelo é treinado diminuindo-se pela metade a resolução das imagens (imagem de entrada de 224x224), voltando à resolução padrão (448x448) para a etapa de detecção [99].

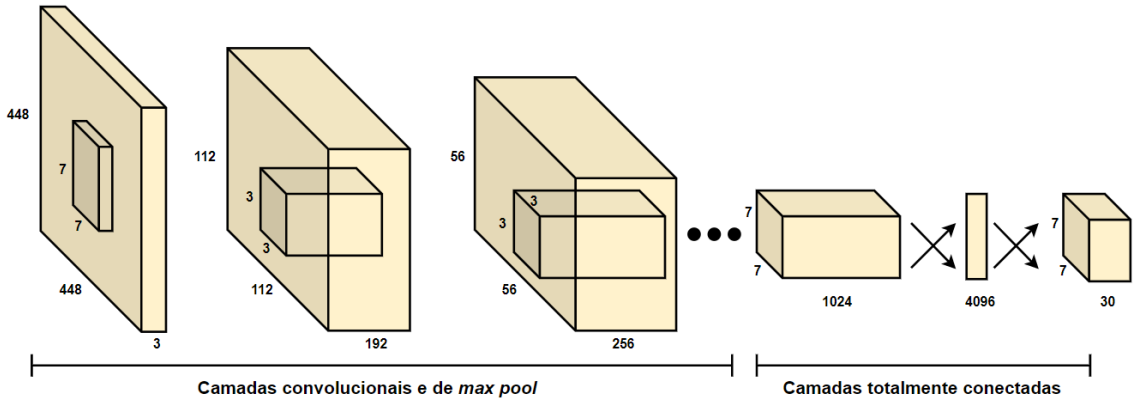



Figura 2.38: Esquema da arquitetura YOLO.

Desde o lançamento da arquitetura de detecção de objetos YOLO variações foram propostas visando melhorias em sua acurácia. A primeira versão, denominada YOLOv1, obteve *mean average precision* (mAP) de 63,4% no desafio PASCAL VOC 2007, enquanto que a versão subsequente (YOLOv2) obteve mAP de 78,6% nessa mesma base de dados. A versão mais recente, YOLOv3, alcançou mAP de 57,9% no banco de imagens COCO (*Common Objects in Context*), superando a versão anterior, que obteve mAP de 48,1% nessa base [100]. Com relação às diferenças arquiteturais, a versão YOLOv2 inovou ao substituir a rede principal do modelo YOLO por uma Darknet-19 e remover as camadas totalmente conectadas presentes no final da arquitetura. A versão mais recente, YOLOv3, difere-se da antecessora por apresentar melhorias em sua rede convolucional. YOLOv3 realiza a predição de áreas retangulares em três escalas diferentes, facilitando a detecção de objetos menores. A rede ficou maior, sendo chamada de Darknet-53, composta de 53 camadas. Essa versão também apresentou aumento da velocidade de processamento, utilizando melhor os recursos de GPU [94].

### 2.5.11 *Data Augmentations*

Uma técnica comumente empregada para evitar o *overfitting* de modelos chama-se *data augmentation*. A baixa quantidade de imagens muitas vezes disponível na etapa de treinamento pode comprometer a capacidade de generalização do sistema desenvolvido. Assim, por meio de transformações específicas, visa-se aumentar artificialmente a quantidade de imagens disponíveis para o treinamento da rede neural convolucional.

Operações como rotações, translações, inclinações, alterações de *zoom* e inserção de ruído são exemplos de *data augmentations* normalmente empregados em projetos de *deep*

*learning* que envolvem a manipulação de imagens. A Figura 2.39 exibe uma imagem original seguida de três imagens obtidas artificialmente por meio da aplicação de *data augmentation*.

Figura 2.39: Imagem original e três imagens artificiais geradas por meio das transformações de mudança de escala, *zoom* e giro horizontal, respectivamente.

### 2.5.12 Reprodutibilidade

Replicar um experimento é executar exatamente as mesmas tarefas que o pesquisador original, com a expectativa de gerar resultados iguais. Em computação científica, por exemplo, a replicação consiste em gerar o mesmo programa com o mesmo compilador utilizando-se os mesmos sistema operacional e *hardware* que o original. De maneira geral, poderá ser difícil replicar todos os aspectos do ambiente original, podendo essas pequenas diferenças acarretar em resultados finais desiguais [102].

A reprodutibilidade é necessária para a verificação dos próprios resultados ou dos obtidos por outro pesquisador, para permitir a extensão do projeto por terceiros, bem como para permitir a manutenção e o suporte por uma comunidade [102]. Em um primeiro momento, experimentos computacionais parecem ser mais fáceis de se reproduzir que experimentos físicos, assumindo-se o computador como uma máquina determinística. Porém, na prática, devido às complexidades dos atuais *hardwares* e *softwares*, é surpreendentemente difícil até mesmo para descrever as entradas de maneira precisa, construir um programa determinístico ou identificar um *hardware* equivalente [102].

Uma pesquisa realizada pela revista *Nature* em 2016 revelou que mais de 70% dos pesquisadores já falharam ao tentar reproduzir experimentos de outros cientistas, enquanto que mais da metade já relatou ter falhado ao tentar reproduzir seus próprios experimentos [103]. Dos 1576 pesquisadores entrevistados, mais da metade (52%) afirmaram existir uma significativa crise de reprodutibilidade, enquanto que 38% consideraram haver uma pequena crise na reprodutibilidade [103].

A crise de reprodutibilidade também está afetando o campo da inteligência artificial (IA). Pesquisadores de IA estão tendo dificuldades em reproduzir muitos resultados chaves, o que está levando a uma nova conscientização acerca de métodos de pesquisas e protocolos

de publicação. Essa dificuldade na reprodutibilidade em IA está relacionada geralmente ao não compartilhamento do código fonte e dos dados utilizados pelo pesquisador.

### 2.5.13 Software Autopsy

Autopsy [104] é um *software* utilizado na perícia computacional o qual fornece uma interface gráfica à coleção de ferramentas de linha de comando *Sleuth Kit*. Esse software é *open source* e pode ser executado em diversas plataformas, além de fornecer a possibilidade de inserção de *plugins* (módulos) desenvolvidos em Java e Python. A tela inicial do Autopsy versão 4.14.0 é exibida na Figura 2.40.


Figura 2.40: Tela inicial do Autopsy versão 4.14.0.

O *software* Autopsy pode executar análises dos tipos *dead analisys* e *live analysis*. A primeira espécie ocorre quando um sistema de análise dedicado é empregado a fim de examinar os dados de um sistema suspeito. Nesse caso, Autopsy e *Sleuth kit* executam em um ambiente confiável. O segundo tipo, *live analysis*, ocorre quando o sistema sob investigação está sendo analisado enquanto em execução. Nesse cenário, Autopsy e *Sleuth kit* podem ser executados a partir de um CD em um ambiente não confiável [105]. Dentre as funcionalidades do Autopsy destacam-se análise de *timeline*, busca por palavras-chaves, filtro por *hash*, *data carving*, etc.

#### Módulos no Autopsy

Ao analisar um caso no Autopsy, o investigador pode selecionar um módulo complementar que será executado na perícia de um dispositivo. Os módulos em Autopsy podem ser de dois tipos: *ingest modules* e *report modules*. O primeiro tipo é executado sempre que uma fonte de dados, como a imagem de um disco rígido em formado *E01* ou uma pasta lógica, por exemplo, são adicionados ao caso sob investigação. O segundo tipo de módulo é

executado após a análise e a seleção dos arquivos de interesse pelo investigador, fornecendo as informações relevantes na forma de um relatório específico [44].

Os módulos *ingest* podem ser divididos em *file ingest modules* e *data source modules*. A principal diferença está no fato de os comandos de um *file ingest module* serem acionados para cada arquivo existente na fonte de dados, ao passo que os *data source ingest modules* recebem a referência a uma fonte de dados, sendo responsabilidade do desenvolvedor localizar os arquivos a serem processados pelo módulo. Independentemente do tipo de módulo a ser empregado, é possível acionar a execução de aplicações externas para a realização de tarefas específicas [44].

# Capítulo 3

# Metodologia

Este Capítulo apresenta a metodologia seguida no desenvolvimento do presente projeto, explorando cada uma das etapas envolvidas na classificação de imagens e detecção de objetos. A metodologia deste trabalho seguiu as etapas propostas no modelo de referência CRISP-DM, apresentado no Capítulo 2. Assim, as próximas seções abordarão as fases de entendimento do negócio, entendimento dos dados, preparação dos dados, modelagem, avaliação e implementação.

## 3.1 Entendimento do negócio

A primeira etapa do modelo de referência CRISP-DM envolve a compreensão dos objetivos do projeto e requerimentos acerca da perspectiva do negócio. A Seção de Perícias de Informática (SPI) da Polícia Civil do Distrito Federal (PCDF) realiza perícias em dispositivos eletrônicos relacionados à prática de crimes propriamente cibernéticos, bem como em equipamentos pertinentes a crimes que se utilizam da informática como ferramenta acessória. Assim, uma ampla gama de delitos é objeto de investigação na presente seção, uma vez que atualmente grande das informações são armazenadas e compartilhadas em meio digital.

Aparelhos celulares, por exemplo, podem armazenar, na forma de imagens, fotografias de possíveis envolvidos em práticas delituosas, veículos utilizados, mapas com locais visitados, e bens adquiridos. Devido ao largo emprego de aplicativos de mensageria, como Whatsapp e Telegram, e de *internet banking*, é frequente o armazenamento de telas contendo conversas privadas e histórico de transferências bancárias. Outro tipo de arquivo de imagem com conteúdo geralmente relevante aos processos investigativos é aquele contendo documentos fotografados, como carteiras de identidade e de habilitação, passaportes, boletos, extratos bancários, contas de energia elétrica, etc. Em computadores pessoais são frequentemente localizadas imagens de documentos escaneados, bem como *backups* de ar-

quivos de celulares, constituindo-se, dessa forma, de uma importante fonte para análise de dados.

Uma das tarefas executadas na seção é a análise e posterior seleção de arquivos possivelmente relevantes ao processo investigativo. Nos crimes de estelionato ou de fraudes tributárias, por exemplo, pode ser relevante localizar arquivos de imagens contendo informações relacionadas a operações financeiras e à identificação de dados pessoais de possíveis autores e vítimas. Nos crimes contra a pessoa e tráfico de drogas, por exemplo, pode ser de interesse à investigação a localização de fotos exibindo indivíduos e/ou armas de fogo.

## 3.2 Entendimento dos dados

Os dados objeto de estudo na etapa de classificação são arquivos de imagem obtidos de casos periciados no âmbito da Seção de Perícias de Informática (SPI) da Polícia Civil do Distrito Federal (PCDF). As imagens foram coletadas de *desktops, notebooks* e *smartphones*, durante os anos de 2018 a 2020, que estavam sob investigação e relacionados a diversos tipos de crimes como estelionato, crimes contra a ordem tributária, homicídios, etc. Foram coletados arquivos de imagem com as extensões *jpg*, *gif*, *bmp* e *png* de dimensões variadas e tamanho a partir de 2 *Kbytes*.

## 3.3 Preparação dos dados

A etapa de preparação dos dados, que envolve coleta, organização e manipulação de imagens, está dividida em duas fases. A *primeira fase* abrange a preparação dos dados que constituirão os conjuntos de treinamento, validação e teste dos classificadores de imagens e do detector de objetos deste projeto. A *segunda fase* compreende a preparação das imagens que serão utilizadas na execução do módulo do *software* Autopsy, o qual empregará os modelos desenvolvidos.

### 3.3.1 Primeira fase

Com base nos tipos de conteúdos de imagens mais frequentes examinados na seção responsável por perícias em dispositivos computacionais e na relevância do conteúdo desses arquivos ao processo investigativo, foram escolhidas três categorias de imagens para a etapa de classificação a serem distribuídas nos grupos de treinamento, validação e teste: *documentos de identificação, documentos gerais* e *não documentos*.

O grupo formado por *documentos de identificação* é constituído de imagens contendo frente e/ou verso de carteira de identidade, motorista, associação de classe ou outra que

exiba informações pessoais (vide Figura 3.1). Essas imagens, contendo a identificação de pessoas, podem estar dispostas de diversas maneiras no arquivo, como em tons de cinza, colorido, ocupando todo o arquivo ou apenas parte dele e exibindo fundo claro.


Figura 3.1: Exemplos de imagens contendo documentos de identificação.

A categoria *documentos gerais* é formada por imagens contendo documentos diversos como boletos de pagamento, extratos bancários, papéis com inscrições manuscritas, *screenshots* de aplicativos de *internet banking* e de mensageria, etc. (vide Figura 3.2). Como principal desafio à etapa de classificação, destaca-se a grande diversidade dessas imagens: documentos podem ocupar totalmente ou apenas parte da imagem, bem como apresentarem *layouts* diversos, dificultando a extração de padrões.

Figura 3.2: Exemplos de imagens contendo documentos gerais.

Por último, a classe *não documentos* é constituída por arquivos de imagem do sistema e outras exibindo desenhos, paisagens, pessoas e objetos diversos (vide Figura 3.3). Das classes de imagens analisadas essa é a mais heterogênea, podendo englobar desde aquivos padrões do sistema operacional a fotografias obtidas por dispositivos móveis, por exemplo.

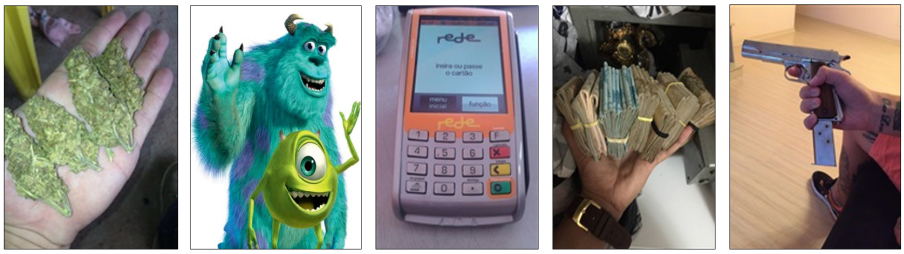



Figura 3.3: Exemplos de imagens do grupo não documentos.

Após a extração dos arquivos presentes em dispositivos sob investigação, com auxílio de *softwares* forenses específicos, foram categorizadas, manualmente, mais de 9.000 imagens de casos periciados ao longo dos anos de 2018 a 2020, as quais foram divididas conforme o disposto na Tabela 3.1.

Tabela 3.1: Arquivos coletados para a etapa de classificação de imagens.

| **Categoria** | **Quantidade** |
|---|---|
| Documentos de identificação | 2300 |
| Documentos gerais | 2300 |
| Não documentos | 4600 |

As imagens, RGB, foram redimensionadas para 128x128 pixels, considerando as limitações da arquitetura utilizada. Foram aplicadas *data augmentations* dos tipos *rescale*, *shear*, *zoom* e *horizontal flip* em cada imagem do conjunto de treinamento de modo a aumentar a variabilidade dos exemplos apresentados aos modelos de classificação. A aplicação dessa técnica gera diferentes variações da imagem em cada época do processamento, assim o total de imagens utilizadas durante a etapa de treinamento é dado pelo número de imagens do conjunto de treinamento multiplicado pelo número de épocas.

Para a fase de detecção de objetos, foram obtidas imagens contendo armas curtas (revólveres e pistolas) e possíveis drogas (maconha) nas formas *in natura*, tabletes e cigarro artesanal. Em virtude da baixa quantidade de imagens contendo esses elementos nos casos periciados, foi necessário complementar o banco de imagens para a detecção de objetos com arquivos obtidos pelos principais mecanismos de busca da *internet*. Termos como "*revólver*", "*pistola*", "*armas curtas*", "*handgun*", "*maconha*", "*marijuana*" e "*weed*" foram fornecidos aos sites de busca *Google* e *Bing*, sendo realizado o *download* manual das imagens mais relevantes apresentadas. O quantitativo de imagens contendo esses elementos está descrito na Tabela 3.2.

Tabela 3.2: Arquivos coletados para a etapa de detecção de objetos.

| **Categoria** | **Quantidade** |
|---|---|
| Armas | 594 |
| Possíveis drogas (maconha) | 224 |

Para o treinamento na fase de detecção de objetos foi necessário, ainda, a anotação manual dos elementos de interesse presentes nas imagens por meio do *software LabelImg* [106]. Após as anotações, é gerado um arquivo *xml* contendo, entre outras informações, o nome do arquivo de imagem, coordenadas da área retangular construída que contém o objeto, bem como sua categoria. Para a geração do arquivo de anotação foi escolhido o

formato PASCAL VOC [107]. Nesta fase de treinamento, as imagens objeto de detecção foram utilizadas em seu tamanho original e não foram empregadas técnicas de *data augmentations*. A Figura 3.4 ilustra um exemplo de anotação de uma imagem contendo o elemento de interesse arma.

```
<size>
    <width>347</width>
    <height>414</height>
    <depth>3</depth>
</size>
<segmented>0</segmented>
<object>
    <name>arma</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
        <xmin>235</xmin>
        <ymin>168</ymin>
        <xmax>325</xmax>
        <ymax>244</ymax>
    </bndbox>
</object>
```

Figura 3.4: Exemplo de anotação de um objeto de interesse e fragmento do arquivo *xml* gerado.

### 3.3.2 Segunda fase

Com os objetivos de exemplificar a utilização e tornar operacionais os classificadores de imagens e o detector de armas desenvolvidos neste projeto, será realizada a integração dos modelos com maiores acurácias ao *software* de perícia computacional Autopsy (Seção 3.4.3).

Para a etapa de classificação foram coletadas 4200 imagens de documentos obtidas de casos sob investigação. Desse quantitativo, 443 imagens são do tipo *documentos de identificação* e 3757 são *documentos gerais*. Além desses arquivos contendo documentos, foram utilizadas 2800 imagens integrantes do conjunto de validação do banco de imagens COCO (*Common Objects in context*) de 2017 [108], totalizando 7000 imagens para análise. Salienta-se que esses arquivos foram coletados após o desenvolvimento dos modelos, não fazendo parte, portanto, dos conjuntos de treinamento, validação e teste. A Tabela 3.3 resume as quantidades de imagens utilizadas nessa etapa.

Tabela 3.3: Quantitativo de imagens utilizado na etapa de classificação.

| **Categoria** | **Quantidade de imagens** |
|---|---|
| Documentos de identificação | 443 |
| Documentos gerais | 3.757 |
| Não documentos | 2.800 |

Para testar a precisão do detector de armas do sistema, foram utilizadas 5000 imagens pertencentes ao conjunto de validação do banco de imagens COCO (*Common Objects in context*) de 2017. Essas imagens contém objetos diversos e estão disponíveis em [108]. Além desse conjunto, foi realizado o *download*, disponível em [109], e utilizadas 2500 imagens contendo armas de fogo. Os quantitativos utilizados estão resumidos na Tabela 3.4.

Tabela 3.4: Quantitativo de imagens utilizado na etapa de detecção.

| **Categoria** | **Quantidade de imagens** |
|---|---|
| Sem armas | 5.000 |
| Com armas | 2.500 |

## 3.4 Modelagem

Após o entendimento do negócio e dos dados, bem como a seleção e preparação dos arquivos que serão objeto de análise, ocorre a fase de modelagem. A parte prática deste projeto está divida em duas etapas principais:

1. Classificação de imagens
2. Detecção de objetos

Na primeira etapa é realizada a *classificação de imagens* utilizando-se uma organização hierárquica de classificadores. O primeiro classificador, denominado *Classificador A*, separará as imagens em dois grupos: *documentos* e *não documentos*. Após a primeira separação, o *Classificador B* atuará sobre os elementos do grupo classificado como *documentos* realizando a sua classificação em *documentos de identificação* e *documentos gerais*, conforme definidos na Seção 3.3.

Na segunda etapa é executada a *detecção de objetos* nas imagens categorizadas como *não documentos*, de acordo com o *Classificador A*. Nesses elementos serão identificadas as presenças de pessoas, automóveis, armas e possíveis drogas. A visão geral das etapas do projeto está apresentada na Figura 3.5.

Essa organização do projeto permite melhor entendimento das etapas envolvidas, bem como auxilia na inspeção e no aperfeiçoamento de classificadores específicos nos casos de possíveis erros de classificação. Outra vantagem dessa organização está em limitar o emprego do classificador B e do detector de objetos apenas em grupos determinados, selecionados pelo primeiro classificador. Documentos de identificação e documentos gerais não apresentam, por padrão, os elementos que serão objeto da etapa de detecção.

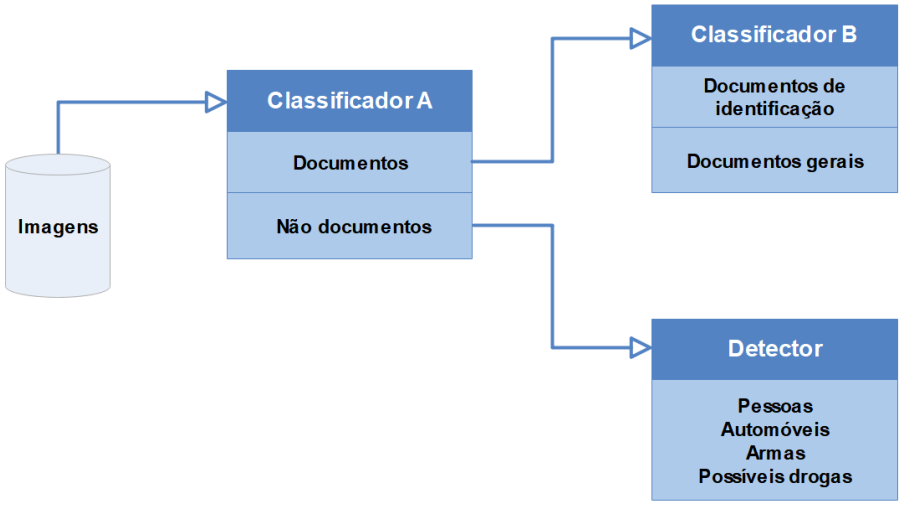



Figura 3.5: Visão geral do projeto.

As etapas de treinamento dos classificadores e dos detectores de objetos deste trabalho foram executadas em um *desktop* no ambiente de trabalho. Trata-se de um equipamento HP Z4 *Workstation* dotado de processador Intel modelo Xeon W-2175, memória RAM de 64GB, disco rígido de 4TB e placa de vídeo marca Nvidia modelo Quadro P4000 (8 GB GDDR5).

### 3.4.1 Classificação de imagens

Para cada um dos classificadores serão utilizados uma *Convolutional Neural Netwok* (CNN) própria e quatro modelos pré-treinados, utilizando-se a técnica de transferência de aprendizado. Para que os resultados sejam menos suscetíveis a variações dos elementos dos conjuntos de treinamento, validação e teste, o processo de classificação da CNN própria será executado cinco vezes, de modo que o treinamento, a validação e o teste do modelo sejam realizados em grupos de arquivos diferentes a cada execução.

Assim, para cada classificador serão gerados nove modelos distintos: cinco gerados a partir da CNN própria (com conjuntos de treinamento, validação e teste diferentes) e quatro modelos obtidos a partir de redes pré-treinadas.

Antes da implementação das arquiteturas que serão usadas no processo de classificação, foi estipulado que os conjuntos de treinamento, validação e teste dos classificadores A e B teriam as distribuições apresentadas nas Tabelas 3.5 e 3.6. Salienta-se que, apesar desse quantitativo fixo, antes do treinamento da CNN desenvolvida, as imagens integrantes dos conjuntos de treinamento, validação e teste são escolhidas de forma aleatória.

Para a etapa de classificação de imagens foi utilizada a linguagem *Python* 3.6, no ambiente de desenvolvimento Anaconda, IDE *Spyder* versão 3.3.1 e *Jupyter Notebook* versão 6.0.3. Para a manipulação dos dados, implementação de algoritmos de *machine*

Tabela 3.5: Distribuição dos arquivos de imagem (*Classificador A*).

| **Categoria** | **Treinamento** | **Validação** | **Teste** |
|---|---|---|---|
| Documentos | 2500 | 1200 | 600 |
| Não documentos | 2500 | 1200 | 600 |

Tabela 3.6: Distribuição dos arquivos de imagem (*Classificador B*).

| **Categoria** | **Treinamento** | **Validação** | **Teste** |
|---|---|---|---|
| Documentos de identificação | 1400 | 600 | 300 |
| Documentos gerais | 1400 | 600 | 300 |

*learning* e exibição de gráficos foram empregadas as bibliotecas *tensorflow, keras, imageai, numpy, pandas* e *seaborn*.

### Convolutional Neural Networks

A CNN foi configurada com o otimizador *Adam* com taxa de aprendizagem (*learning rate*) padrão de 0,001 e o processo de treinamento compreendeu 50 épocas com *batch* de tamanho 32. A função de perda utilizada foi a *categorical cross-entropy*, cujo valor retornado será uma probabilidade entre 0 e 1. Salienta-se que as CNN dos dois classificadores apresentam as mesmas configurações, como número e tipos de camadas, funções de ativação, variáveis de treinamento, etc.

A arquitetura apresenta *kernels* de tamanho 3x3 e *pools* 2x2, ao passo que as funções de ativação das camadas convolucionais são do tipo ReLu e a da camada final é do tipo *sigmoid*. Detalhes da arquitetura da CNN implementada estão dispostos na Tabela 3.7. Os dois primeiros elementos da tupla do formato de saída, presente na tabela, indicam as dimensões de entrada, enquanto que o último valor da tupla fornece o número de filtros utilizados.

A escolha desses hiperparâmetros ocorreu variando-se os números de épocas, de camadas, os tamanhos e quantidades de filtros, as dimensões da imagem de entrada e o tamanho do *batch*. Essa seleção de parâmetros objetivou criar modelos com a maior acurácia possível, considerando-se as limitações da arquitetura utilizada. Salienta-se que, devido à confidencialidade dos dados presentes nas imagens manipuladas, não foi possível submeter os arquivos a ambientes de processamento mais robustos. Dos cenários testados, a configuração apresentada foi a que produziu os melhores resultados.

### Transferência de aprendizado (*Transfer Learning*)

Os modelos VGG16, VGG19 (*Very Deep Convolutional Networks for Large ScaleImage Recognition*), Xception e InceptionV3, pré-treinados para o desafio de classificação de ima-

Tabela 3.7: Arquitetura do classificador A e do classificador B.

| **Tipo de camada** | **Formato de saída** | **Nº de parâmetros** |
|---|---|---|
| Conv2D | (126, 126, 64) | 1.792 |
| MaxPooling | (63, 63, 64) | 0 |
| Conv2D | (61, 61, 64) | 36.928 |
| MaxPooling | (30, 30, 64) | 0 |
| Conv2D | (28, 28, 128) | 73.856 |
| MaxPooling | (14, 14, 128) | 0 |
| Conv2D | (12, 12, 128) | 147.584 |
| MaxPooling | (6, 6, 128) | 0 |
| Conv2D | (4, 4, 256) | 295.168 |
| MaxPooling | (2, 2, 256) | 0 |
| Flatten | (1024) | 0 |
| Dense | (256) | 262.400 |
| Dense | (2) | 514 |

gens *ImageNet*, foram utilizados neste projeto. Os pesos presentes nos modelos obtidos a partir da classificação *ImageNet* foram mantidos, alterando-se apenas as últimas camadas de modo a adaptar o modelo ao problema de classificação específico deste trabalho. Todos os modelos receberam como entradas arquivos de imagem de tamanho 128x128 *pixels*, com 3 canais, e foram treinados por 50 épocas. A exemplo da CNN descrita no item anterior, os modelos implementados também empregaram a função de ativação *categorical cross-entropy*.

A Tabela 3.8 apresenta as principais características padrões dos modelos disponíveis na biblioteca *keras* e utilizados para *transfer learning*. A seguir serão descritas as principais alterações realizadas nesses modelos, a fim de adapta-los a este projeto.

Tabela 3.8: Características dos modelos escolhidos.

| **Modelo** | **Tamanho** | **Top-1 Accuracy** | **Top-5 Accuracy** | **Parâmetros** | **Profundidade** |
|---|---|---|---|---|---|
| VGG16 | 528 mb | 0,713 | 0,901 | 138.357.544 | 23 |
| VGG19 | 549 mb | 0,713 | 0,900 | 143.667.240 | 26 |
| Xception | 88 mb | 0,79 | 0,945 | 22.910.480 | 126 |
| InceptionV3 | 92 mb | 0,779 | 0,937 | 23.851.784 | 159 |

Os modelos com as configurações descritas na Tabela 3.8 foram adaptados mantendo-se suas camadas iniciais pré-treinadas, porém adicionando-se uma camada totalmente conectada ao final, com dois nós na saída (número de classes) e função sigmoide. As arquiteturas resultantes estão expostas nas Figuras 3.9 a 3.12.

Tabela 3.9: Arquitetura adaptada do modelo VGG16 utilizada para transferência de aprendizado.

| **Tipo de camada** | **Formato de saída** | **Nº de parâmetros** |
|---|---|---|
| Entrada | (128, 128, 3) | 0 |
| Modelo VGG16 | (4, 4, 512) | 14.714.688 |
| Flatten | (8192) | 0 |
| Dense | (2) | 16.386 |

Tabela 3.10: Arquitetura adaptada do modelo VGG19 utilizada para transferência de aprendizado.

| **Tipo de camada** | **Formato de saída** | **Nº de parâmetros** |
|---|---|---|
| Entrada | (128, 128, 3) | 0 |
| Modelo VGG19 | (4, 4, 512) | 20.024.384 |
| Flatten | (8192) | 0 |
| Dense | (2) | 16.386 |

Tabela 3.11: Arquitetura adaptada do modelo Xception utilizada para transferência de aprendizado.

| **Tipo de camada** | **Formato de saída** | **Nº de parâmetros** |
|---|---|---|
| Entrada | (128, 128, 3) | 0 |
| Modelo Xception | (4, 4, 2048) | 20.861.480 |
| Flatten | (32.768) | 0 |
| Dense | (2) | 65.538 |

### 3.4.2 Detecção de pessoas e objetos

A fase de detecção de elementos desse projeto envolve a detecção de pessoas, automóveis, armas de fogo e possíveis drogas (maconha). Como os modelos pré-treinados utilizados já realizam a detecção de pessoas e automóveis, os processos de anotação e treinamento envolveram apenas as imagens contendo armas e possíveis drogas.

Após a etapa de preparação dos dados, que compreendeu a coleta, organização e anotação manual dos objetos de interesse, foi realizada a distribuição dos arquivos de imagem nos conjuntos de treinamento e validação. Os quantitativos de cada grupo estão descritos na Tabela 3.13.

Para a detecção de objetos foi utilizada a linguagem *Python* 3.6, no ambiente de desenvolvimento Anaconda, e IDE *Jupyter Notebook* versão 6.0.3. Para o treinamento e uso do algoritmo YOLO foi empregada a biblioteca *ImageAI* [110]. Pra fins de comparação, foram realizados dois treinamentos: o primeiro utilizando o modelo pré-treinado

Tabela 3.12: Arquitetura adaptada do modelo InceptionV3 utilizada para transferência de aprendizado.

| **Tipo de camada** | **Formato de saída** | **Nº de parâmetros** |
|---|---|---|
| Entrada | (128, 128, 3) | 0 |
| Modelo InceptionV3 | (4, 4, 2048) | 21.802.784 |
| Flatten | (8192) | 0 |
| Dense | (2) | 16.386 |

Tabela 3.13: Distribuição dos arquivos para a detecção de objetos.

| **Categoria** | **Treinamento** | **Validação** |
|---|---|---|
| Armas | 396 | 198 |
| Possíveis drogas (maconha) | 159 | 65 |

YOLOv3 e o segundo sem a utilização desse modelo pré-treinado. Em cada um desses casos, o processo de treinamento ocorreu por 100 épocas, demandando cerca de 20 horas de processamento no computador com as especificações descritas na Seção 3.4. Considerando as limitações de memória do ambiente utilizado, foi escolhido *batch* de tamanho 4.

## 3.5 Avaliação

As avaliações dos modelos de classificação de imagens e detecção de objetos estão presentes no Capítulo Resultados. Serão apresentadas as acurácias obtidas pelos modelos na classificação dos elementos do conjunto de validação e teste, bem como métricas para a detecção de objetos.

## 3.6 Implementação

A fim de tornar operacional, no ambiente pericial, os classificadores e detector de elementos em imagens deste projeto foram integrados ao *software* Autopsy, apresentado na Seção 2.5. Ao oferecer suporte à inserção de *plugins* desenvolvidos nas linguagens de programação Java e Jython (uma versão de Python para a máquina virtual Java), Autopsy possibilita o incremento de suas funcionalidades, auxiliando na solução de problemas de ordem prática na perícia computacional. A Figura 3.6 exibe o processo de integração dos modelos gerados neste projeto ao *software* Autopsy.

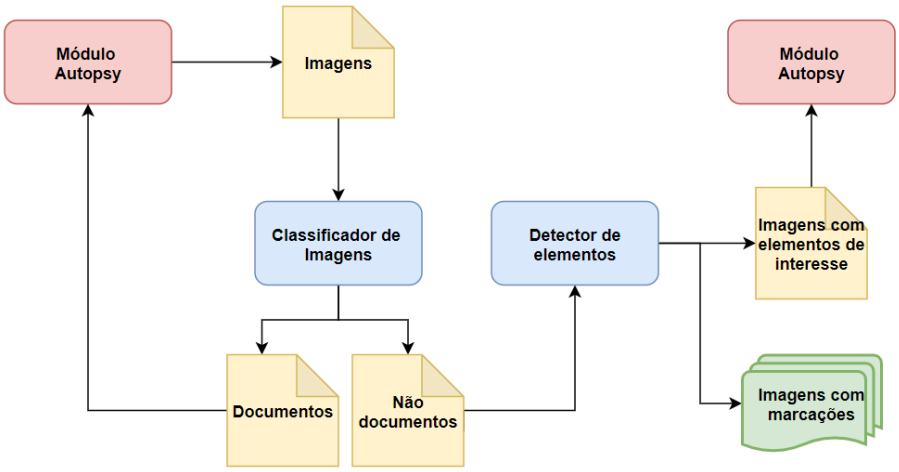



Figura 3.6: Estrutura dos módulos desenvolvidos para o Autopsy.

### Módulo classificador de imagens

Após a etapa de classificação deste projeto, foi desenvolvido um módulo do tipo *data source ingest* a fim de oferecer a seleção de imagens contendo documentos de identificação e documentos gerais ao Autopsy. Por padrão, a classificação de imagens contemplará arquivos com as extensões *png*, *jpg*, *bmp* e *gif*.

A execução do módulo inicia pela pesquisa por arquivos, contendo as extensões de interesse, presentes nas fontes de dados sob investigação. Após essa pesquisa, é criado um arquivo de texto contendo o nome, a extensão e o diretório das imagens a serem classificadas. Assim, depois dessas etapas a aplicação *image_classifier.exe*, responsável pela classificação de imagens, é acionada pelo código Jython que fornece como parâmetro o *path* do arquivo de texto ao executável.

A aplicação de linha de comando *image_classifier.exe* é responsável por carregar os modelos contendo as CNN treinadas para a classificação de imagens (apresentadas na Seção 3.4.1), percorrer o arquivo passado como parâmetro, fornecer os arquivos de imagem aos classificadores e analisar seus resultados. Por fim, a aplicação de linha de comando gera dois arquivos contendo os nomes, extensões e diretórios das imagens classificadas como documentos de identificação e documentos gerais.

Quando a aplicação *image_classifier.exe* é finalizada, a execução retorna ao módulo do Autopsy. Nessa última etapa, os arquivos gerados pela aplicação são analisados pelo código Jython e cada imagem descrita é registrada sob a forma de artefato (*artifact*) em Autopsy. Esses artefatos são reunidos em dois grupos denominados *Documentos_ID*, contendo as imagens classificadas como documentos de identificação, e *Documentos_gerais*, exibindo as imagens assim classificadas. Essas classificações podem ser acessadas por meio da interface gráfica da ferramenta, estando à disposição do investigador para a seleção das

imagens de interesse. Assim, com a integração do classificador de imagens deste projeto, a ferramenta oferece a possibilidade de classificação de imagens nas categorias documentos de identificação e documentos gerais. Essa integração permite a operacionalização do projeto em atividades periciais contribuindo na análise mais célere de arquivos de imagem.

### Módulo detector de pessoas e objetos

A etapa de detecção de objetos permitiu a criação de modelo para a detecção de armas o qual foi integrado ao *software* Autopsy. Além da detecção de armas, a arquitetura YOLOv3, por meio da biblioteca *ImageAI* [110], oferece o reconhecimento de diversos objetos em imagens. Assim, optou-se por integrar a detecção de pessoas, carros e armas a este módulo, pois esses elementos são, geralmente, de interesse durante perícias computacionais.

O funcionamento do módulo para detecção de pessoas e objetos ocorre de modo análogo ao de classificação de imagens, sendo inicialmente fornecido uma lista de arquivos de imagens, com as extensões de interesse, a uma aplicação externa denominada *object_detector.exe*. Após fornecer as imagens ao modelo de detecção de pessoas e objetos, a aplicação gera um arquivo *csv* contendo as classes do elementos localizados (pessoa, carro ou arma) e os *paths* do arquivos que contêm esses objetos.

Esse arquivo de extensão *csv* é, então, lido pelo módulo do Autopsy e as imagens contendo pessoas, carros ou armas são agrupadas nas categorias correspondentes no painel de visualização. Como as imagens objeto de perícia devem estar inalteradas, é criado um diretório contendo as cópias das imagens com os elementos de interesse envoltos em uma área retangular, indicando a posição desse objeto ou pessoa. Além dessa marcação gráfica, também é fornecido o percentual retornado pelo modelo na detecção do objeto. A configuração inicial permite ao modelo identificar e assinalar um objeto sempre que a probabilidade retornada for superior a 50%.

Os módulos de classificação de imagens e detecção de pessoas e objetos podem atuar de forma integrada, em que há a detecção de elementos apenas nos arquivos classificados como *não documentos*, ou de forma separada, na qual a detecção ocorre em todos os arquivos com as extensões definidas. Salienta-se que o *plugin* desenvolvido semi-automatiza os processos de classificação e detecção, uma vez que não é dispensada a validação e a seleção manual dos arquivos de imagem selecionados pelo módulo, de acordo com os objetivos periciais. A Figura 3.7 exibe o painel de navegação do Autopsy destacando as categorias criadas pelos módulos após a análise de um caso.

Para fins de simplificação do projeto e considerando os elementos (pessoas, carros e armas) geralmente mais relevantes às investigações realizadas, o detector analisará, sucessivamente, a presença de armas, pessoas e veículos em imagens. Uma vez detectado

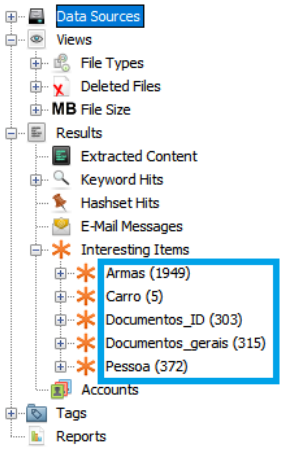



Figura 3.7: Painel do Autopsy exibindo as categorias documentos de identificação e documentos gerais, bem como os grupos de imagens contendo pessoas, carros e armas.

um desses elementos, a imagem original objeto de análise será classificada na categoria correspondente, bem como será criada uma cópia da imagem contendo o objeto identificado em destaque. Assim, por exemplo, uma fotografia contendo uma pessoa portando uma arma de fogo seria classificada na categoria “Armas” do painel do Autopsy e não nas categorias “Armas” e “Pessoas” simultaneamente.

# Capítulo 4

# Resultados

Este Capítulo apresenta os resultados obtidos neste projeto, sendo descritos os desempenhos dos modelos, em termos de acurácias, para a classificação de imagens e detecção de objetos. Também é discutido o processo de classificação por meio da análise visual das ativações ocorridas na rede neural convolucional.

## 4.1 Classificação de imagens

Os resultados da etapa classificação de imagens, formada pelo classificador A, responsável pela separação entre *documentos* e *não documentos*, e pelo classificador B, responsável pela classificação entre *documentos de identificação* e *documentos gerais*, estão descritos nesta Seção. As métricas empregadas estão definidas na Seção 2.4.10.

O processo de treinamento dos classificadores A e B demandou cerca de 7,5 horas e, ao final dessa etapa, o classificador A desenvolvido por meio de uma rede neural convolucional própria (sem o uso de *transfer learning*) apresentou acurácia de 0,982 na classificação das imagens do conjunto de teste. O classificador B atingiu acurácia de 0,987 na classificação dos elementos integrantes do conjunto de teste. Os resultados obtidos por esses classificadores, juntamente com os fornecidos pelos modelos que utilizaram transferência de aprendizado, estão descritos nas Tabelas 4.1 e 4.2. A Figura 4.1 exibe as matrizes de confusão com os resultados obtidos pelos classificadores A e B, respectivamente. Para a geração dessa matriz foi considerada a classificação das imagens integrantes do conjunto de teste.

A Figura 4.2 exibe exemplos de falso-positivos para documentos (a) e não documentos (b) indicados pelo classificador A. Analisando as imagens equivocadamente classificadas como documentos, constata-se que esses arquivos são geralmente pequenos (aproximadamente 4 *KBytes*) e possuem fundo de tonalidade clara, assemelhando-se a documentos. Já no grupo de imagens erroneamente categorizadas como não documentos estão, por

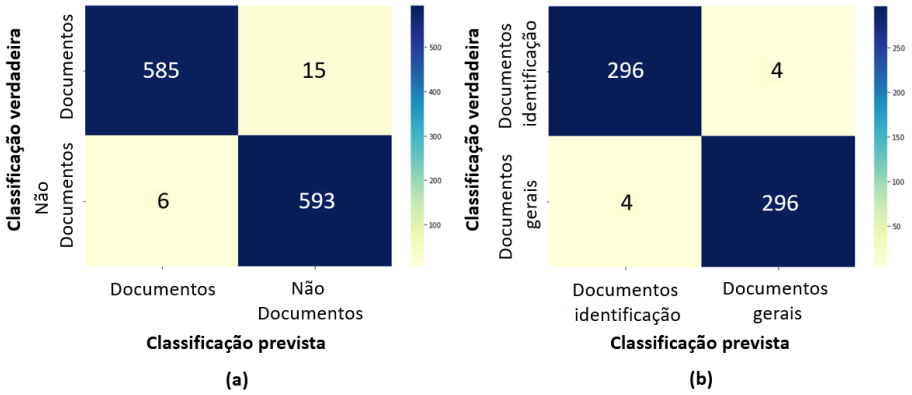



Figura 4.1: Matrizes de confusão dos classificadores A e B.

exemplo, arquivos parcialmente ilegíveis, imagens com erro na anotação manual inicial e arquivos contendo papéis fotografados sobre superfícies diversas, como mesas.

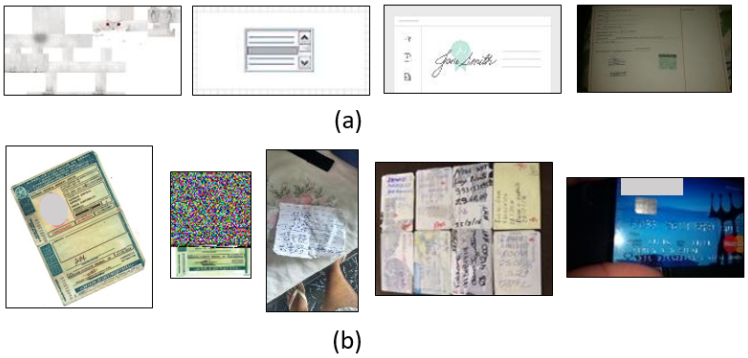



Figura 4.2: Exemplos de falso-positivos para documentos (a) e não documentos (b).

A Figura 4.3 exibe exemplos de falso-positivos para documentos de identificação (a) e documentos gerais (b) indicados pelo classificador B. Dos falso-positivos para documentos de identificação há dois certificados de registro e licenciamento de veículo e um comprovante de pagamento escaneado. A provável origem do erro de classificação para esses tipos de imagens será explorada na Seção 4.2. Arquivos contendo carteiras de classe profissional ocupando apenas parte da imagem, por exemplo, integram o conjunto de falso-positivos para documentos gerais, conforme exposto na Figura 4.3 (b).

Além da CNN desenvolvida, foram utilizados os modelos pré-treinados VGG16, VGG19, Xception e InceptionV3 para os classificadores A e B deste projeto. O processo de treinamento de cada modelo durou cerca de 1h15min (50 épocas), sendo usados os mesmos conjuntos de treinamento, validação e teste em cada modelo pré-treinado. A Figura 4.4 exibe a evolução dos valores retornados pela função de perda ao longo dos treinamentos em que foi utilizado transferência de aprendizado.

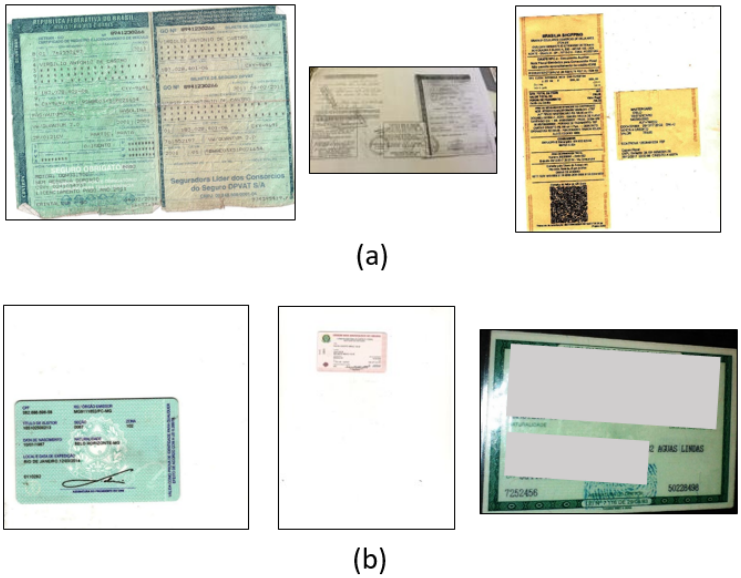



Figura 4.3: Exemplos de falso-positivos para documentos de identificação (a) e documentos gerais (b).

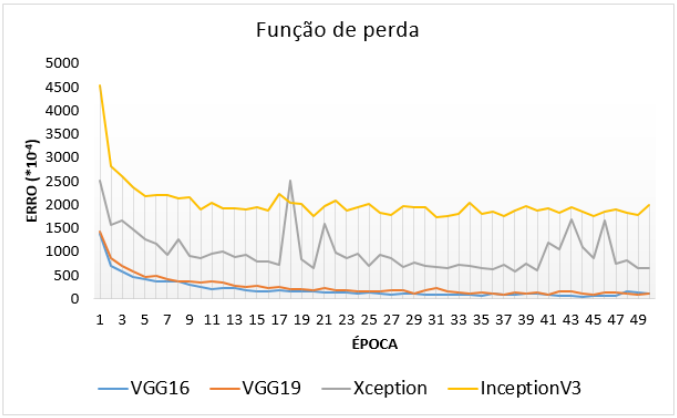



Figura 4.4: Valores retornados pela função de perda durante o treinamento dos modelos com transferência de aprendizado.

Os resultados obtidos por todos os modelos gerados estão apresentados nas Tabelas 4.1 e 4.2. São exibidas as principais métricas alcançadas na classificação dos elementos dos conjuntos de validação e teste na classificação entre documentos e não documentos (*Classificador A*), bem como na classificação entre documentos de identificação e documentos gerais (*Classificador B*). As matrizes de confusões dos modelos que empregam transferência de aprendizado, ilustradas nas Figuras 4.5 a 4.8, contém os resultados obtidos na classificação dos elementos do conjunto de teste.

Tabela 4.1: Resultados do classificador A.

| | Conjunto de validação | | | Conjunto de teste | | |
|---|---|---|---|---|---|---|
| | **Acurácia** | **Precision** | **F1 score** | **Acurácia** | **Precision** | **F1 score** |
| **VGG16** | 0,986 | 0,983 | 0,987 | 0,982 | 0,979 | 0,982 |
| **VGG19** | 0,986 | 0,986 | 0,986 | 0,986 | 0,98 | 0,976 |
| **CNN própria** | 0,979 | 0,981 | 0,979 | 0,982 | 0,99 | 0,982 |
| **InceptionV3** | 0,914 | 0,857 | 0,921 | 0,91 | 0,85 | 0,917 |
| **Xception** | 0,945 | 0,91 | 0,948 | 0,94 | 0,9 | 0,943 |

Tabela 4.2: Resultados do classificador B.

| | Conjunto de validação | | | Conjunto de teste | | |
|---|---|---|---|---|---|---|
| | **Acurácia** | **Precision** | **F1 score** | **Acurácia** | **Precision** | **F1 score** |
| **VGG16** | 0,997 | 0,995 | 0,997 | 0,997 | 0,997 | 0,997 |
| **VGG19** | 0,998 | 0,998 | 0,998 | 0,998 | 1 | 0,998 |
| **CNN própria** | 0,978 | 0,985 | 0,978 | 0,987 | 0,987 | 0,987 |
| **InceptionV3** | 0,856 | 0,976 | 0,838 | 0,88 | 0,98 | 0,867 |
| **Xception** | 0,844 | 0,937 | 0,829 | 0,848 | 0,956 | 0,828 |

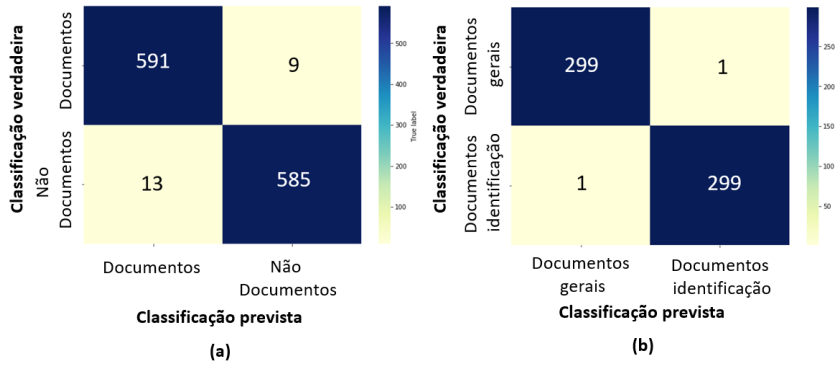



Figura 4.5: Matrizes de confusão dos classificadores A e B, respectivamente, geradas utilizando-se o modelo VGG16.

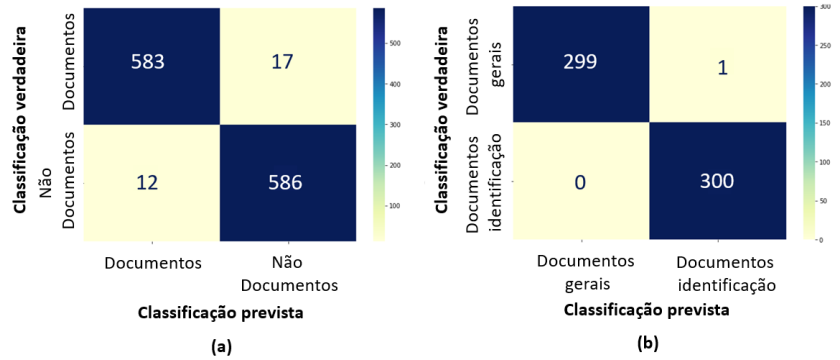



Figura 4.6: Matrizes de confusão dos classificadores A e B, respectivamente, geradas utilizando-se o modelo VGG19.

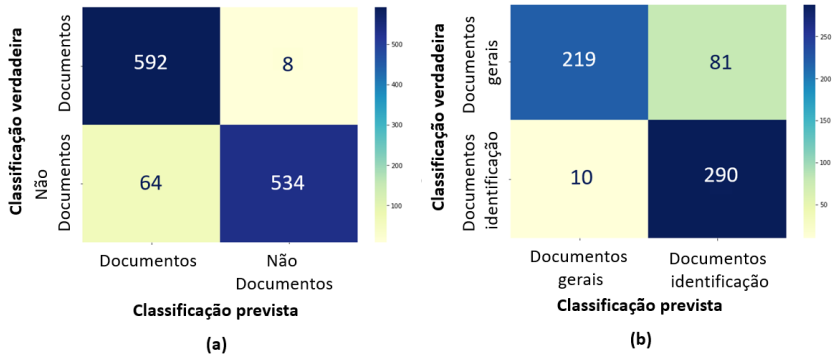



Figura 4.7: Matrizes de confusão dos classificadores A e B, respectivamente, geradas utilizando-se o modelo Xception.

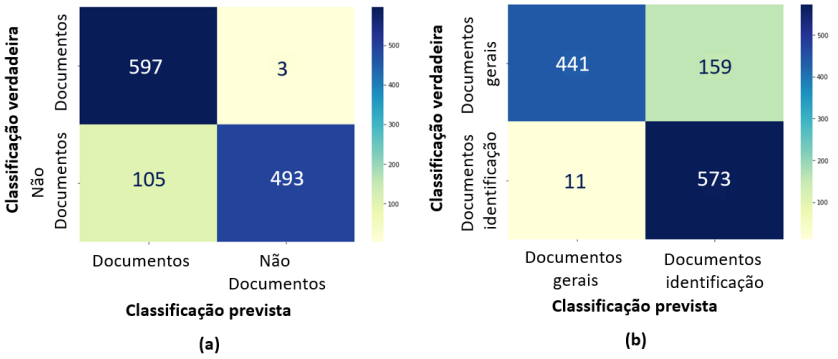



Figura 4.8: Matrizes de confusão dos classificadores A e B, respectivamente, geradas utilizando-se o modelo InceptionV3.

## 4.2 Visualização e análise da CNN no processo de classificação de imagens

Essa Seção visa ilustrar o processo de aprendizagem ocorrido na rede neural convolucional implementada. As representações aprendidas pelas CNNs são favoráveis às visualizações porque são representações de conceitos visuais [111]. As exibições apresentadas serão divididas em *visualizações de saídas intermediárias da CNN*, *visualização de filtros* e *visualizações de mapas de calor da ativação de uma classe em uma imagem*, conforme estudadas em [111].

A visualização de ativações intermediárias consiste na exibição de *feature maps* que são saídas de várias camadas de convolução e *pooling* em uma rede, dada uma imagem. Isso fornece uma visualização de como a entrada está sendo decomposta em diferentes filtros aprendidos pela rede [111].

A Figura 4.9 (a) exibe um exemplo de imagem de uma carteira nacional de habilitação fornecida ao classificador B, cuja função é classificar uma imagem em *documento de identificação* ou em um *documento geral*. A Figura 4.9 (b) apresenta, como exemplo, o

sétimo canal do *feature map* da primeira camada do modelo. A figura sugere que esse canal atua como detector de linhas verticais ou diagonais.

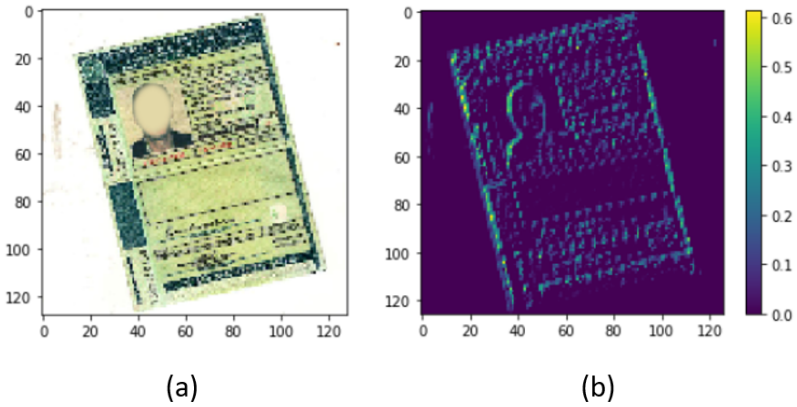



Figura 4.9: Exemplos de imagem de entrada e *feature map* obtido.

Além de canais individuais é possível analisar o conjunto de todos os canais presentes no primeiro *activation map* da rede neural convolucional do classificador B. A partir da análise dos canais apresentados na Figura 4.10, verifica-se uma tendência de detecção de contornos gerais, como as bordas da carteira de habilitação. Nesse estágio, as ativações retêm praticamente toda a informação presente na imagem inicial [111]. As *features* extraídas por uma camada tornam-se mais abstratas à medida em que se aumenta a profundidade da camada [111].

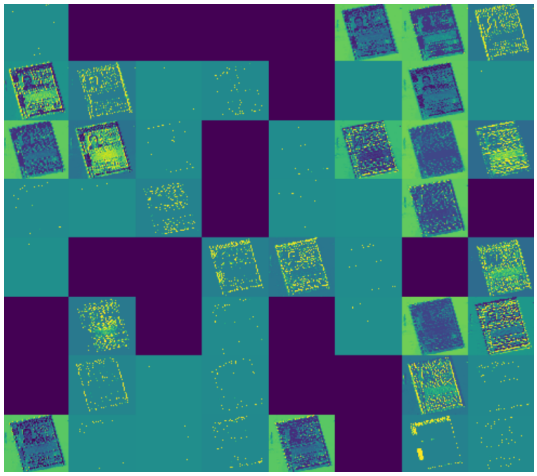


Figura 4.10: Exemplo de conjunto de *feature maps*.

Assim como é possível analisar as *features maps* geradas, pode-se inspecionar os filtros utilizadas para a formação dos canais. A Figura 4.11 exibe exemplos de filtros utilizados na segunda camada de convolução da CNN do classificador B.

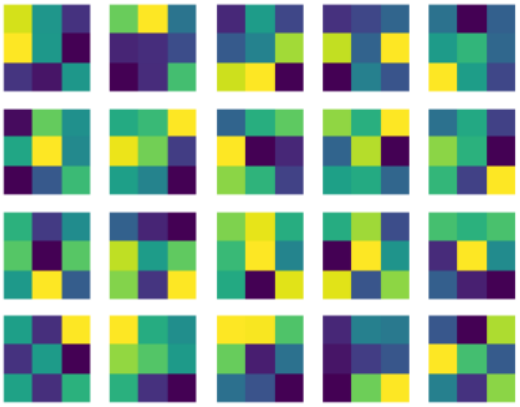


Figura 4.11: Exemplos de filtros.

Prosseguindo as análises acerca do processo de aprendizado da CNN, é relevante observar que determinadas regiões de uma dada imagem tendem a influenciar mais a rede neural convolucional na sua decisão de classificação final. Esse tipo de visualização é denominado *class activation map* (CAM) e consiste na produção de mapas de calor de ativação da classe em uma imagem. Esse mapa de ativação consiste em *grid* de duas dimensões com valores associados à saída específica de uma classe, computada para cada região de uma imagem de entrada, indicando quão importante é essa região para a classificação da classe sob análise [111].

A CAM foi aplicada neste projeto a fim de identificar possíveis regiões foco de ativação durante o processo de classificação entre documentos gerais e documentos de classificação, realizado pelo classificador B. Verificaram-se formações de padrões específicos ao se fornecerem imagens contendo carteiras de habilitação e imagens contendo documentos de identificação diversos quando esses últimos elementos estão em fundo branco.

Quando imagens formadas unicamente por carteiras de habilitação são fornecidas, ou seja, quando o documento ocupa totalmente a imagem, há uma ativação maior da região da borda superior e na área da foto do proprietário do documento, indicando que essas áreas possuem grande relevância quando se trata da classificação dessa classe de documento. Exemplos de imagens apresentando esse padrão estão exibidos na Figura 4.12 (os dados pessoais presentes nas imagens desta Seção foram ocultados).

Ao utilizar o classificador B em imagens contendo documentos de identificação em fundos claros, como por exemplo a fotocópia de uma carteira de identidade digitalizada, há maior ativação no contorno do documento. Assim, as bordas dos documentos de identificação são regiões de destaque quando se classifica esse tipo de imagem. Exemplos de mapas de calor de imagens apresentando esse padrão estão expostos na Figura 4.13.

Esse tipo de análise pode ser bastante útil ao se inspecionar imagens equivocadamente classificadas, por exemplo. A Figura 4.14 exibe falso-positivos para documentos de iden-


Figura 4.12: Padrão de ativação verificado na classificação de documentos de identificação (carteira de habilitação).


Figura 4.13: Padrão de ativação verificado na classificação de documentos (carteira de identidade).

tificação após atuação do classificador B. Ao se analisar, em especial, os certificados de registro e licenciamento de veículo presentes nos itens (a) e (b) constata-se maior ativação em torno da borda superior do documento. Essa área de ativação explica o erro do modelo ao classificar um documento geral em um documento de identificação, uma vez que bordas com características visuais semelhantes também são encontradas em documentos de identificação (carteira nacional de habilitação).

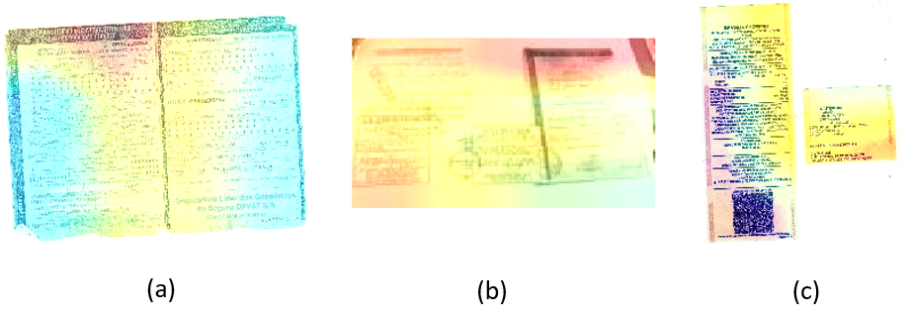



Figura 4.14: Áreas de ativação nos falso-positivos para documentos de identificação classificados pelo classificador B.

## 4.3 Detecção de objetos

Após a realização das etapas descritas no Capítulo Metodologia, modelos para detecção de objetos foram criados a fim de identificar armas e possíveis drogas contidas em imagens objeto de perícia criminal. Para a etapa de detecção de objetos foi empregada a biblioteca *ImageAI*, que fornece a possibilidade de treinamento utilizando-se a rede pré-treinada YOLOv3.

O gráfico contendo a evolução dos valores da função de perda gerados durante o treinamento das redes pré-treinada e não pré-treinada durante a etapa de treinamento está exposto na Figura 4.15.


Figura 4.15: Resultados da função de perda durante a etapa de treinamento (detecção de armas).

Após o treinamento e consequente geração de modelos, foram analisadas suas acurácias, em termos de mAP. Para essa etapa foram fornecidos os valores 0.5, 0.6 e 0.5, respectivamente, para os parâmetros *iou_threshold*, *object_threshold* e *nms_threshold*. O parâmetro *iou_threshold* indica o valor de IoU que será empregado para fins de cálculo da métrica mAP: quanto mais alto for esse valor, maior será a precisão do sistema. O parâmetro *object_threshold* indica o valor mínimo para a probabilidade da classe que será considerada para fins de cálculo do mAP. O último termo, a variável *nms_threshold*, refere-se ao valor mínimo para *Non-maximum Suppression* que será usado na avaliação do mAP. Um detector geralmente possui respostas positivas para múltiplas janelas ao lado da verdadeira marcação do objeto, assim por meio da *Non-maximum Suppression* é possível diminuir essa quantidade excessiva de detecções [93].

No treinamento com o emprego da rede pré-treinada YOLOv3 foram gerados 24 modelos, obtendo-se mAP máximo de 0,85. Ao não se empregar o pré-treinamento, os modelos com menor valor da função de perda e maior mAP obtiveram resultados de 5,82 e 0,59,

respectivamente. Os valores da função de perda e do mAP para cada modelo de detecção de armas gerado ao se utilizar o pré-treinamento estão apresentados na Tabela 4.3.

Tabela 4.3: Valores da função de perda e mAP para cada modelo gerado (detecção de armas).

| **Época** | **Valor da função de perda** | **mAP** | **Época** | **Valor da função de perda** | **mAP** |
|---|---|---|---|---|---|
| **01** | 25,426 | 0,60 | **15** | 3,840 | 0,83 |
| **02** | 9,872 | 0,65 | **16** | 3,433 | 0,85 |
| **03** | 8,224 | 0,77 | **17** | 3,120 | 0,84 |
| **04** | 7,278 | 0,54 | **19** | 2,956 | 0,84 |
| **05** | 6,959 | 0,73 | **21** | 2,941 | 0,83 |
| **06** | 6,344 | 0,77 | **22** | 2,797 | 0,83 |
| **07** | 6,188 | 0,69 | **24** | 2,725 | 0,84 |
| **08** | 5,999 | 0,72 | **27** | 2,707 | 0,83 |
| **09** | 5,592 | 0,65 | **28** | 2,591 | 0,83 |
| **10** | 5,124 | 0,76 | **31** | 2,544 | 0,81 |
| **11** | 5,097 | 0,81 | **32** | 2,495 | 0,82 |
| **12** | 4,696 | 0,82 | **76** | 2,383 | 0,82 |

Para a criação do detector de possíveis drogas em imagens foram utilizadas as mesmas bibliotecas e estrutura apresentadas na detecção de armas. O processo de treinamento compreendeu 200 épocas utilizando-se o modelo pré-treinado YOLOv3. Cada época exigiu cerca de 38 minutos de execução, sendo fornecido, ao final, o valor da função de perda. Devido à pouca variação no valor retornado pela função na segunda metade do treinamento, a Figura 4.16 exibe apenas os resultados retornados pela função de perda para as 100 primeiras épocas.

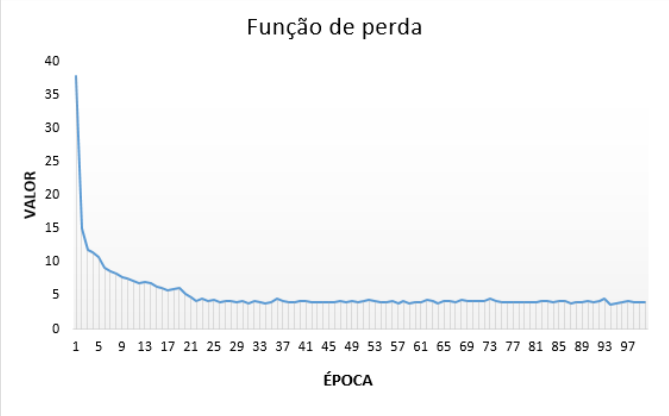



Figura 4.16: Resultados da função de perda durante a etapa de treinamento (detecção de possíveis drogas).

No processo de treinamento do detector para possíveis drogas, foram criados 24 modelos os quais foram avaliados com base na métrica mAP. Os valores da função de perda e da mAP para cada modelo gerado estão descritos na Tabela 4.4. A partir da análise desses valores, observa-se que o modelo com maior precisão obteve mAP de 0,68.

Tabela 4.4: Valores da função de perda e mAP para cada modelo gerado (detecção de possíveis drogas).

| **Época** | **Valor da função de perda** | **mAP** | **Época** | **Valor da função de perda** | **mAP** |
|---|---|---|---|---|---|
| **01** | 37,87 | 0,18 | **14** | 6,79 | 0,62 |
| **02** | 15,06 | 0,15 | **15** | 6,41 | 0,57 |
| **03** | 11,96 | - | **16** | 6,24 | 0,54 |
| **04** | 11,42 | 0,49 | **17** | 5,89 | 0,63 |
| **05** | 10,72 | 0,24 | **20** | 5,26 | 0,59 |
| **06** | 9,17 | 0,63 | **21** | 4,79 | 0,6 |
| **07** | 8,7 | 0,51 | **22** | 4,25 | 0,6 |
| **08** | 8,34 | 0,52 | **26** | 4,06 | 0,65 |
| **09** | 7,72 | 0,64 | **29** | 3,99 | 0,65 |
| **10** | 7,6 | 0,51 | **31** | 3,89 | 0,63 |
| **11** | 7,15 | 0,68 | **34** | 3,8 | 0,63 |
| **12** | 6,95 | 0,6 | **94** | 3,65 | 0,64 |

Após essas avaliações, o modelo com mAP mais elevado (0,68) foi testado a fim de reconhecer possíveis drogas em imagens presentes no conjunto de validação. Dos diferentes tipos de imagens passados, o modelo apresentou maior precisão na identificação de cigarros artesanais, conforme ilustrado na Figura 4.17. Devido ao baixo quantitativo de imagens contendo possíveis drogas obtido, aliado à diversidade de formas desses elementos de interesse (maconha na forma *in natura*, em tabletes ou na forma de cigarro artesanal) o treinamento gerou, de forma geral, modelos menos precisos para o reconhecimento de possíveis drogas. Assim, optou-se por não incluir, nesse momento, esse tipo de reconhecimento ao módulo de detecção de objetos do *software* Autopsy, conforme será apresentado na Seção 4.4.

## 4.4 Execução do módulo Autopsy no ambiente de trabalho

Com os objetivos de exemplificar a utilização e tornar operacionais os classificadores de imagens e o detector de armas desenvolvidos neste projeto, foi realizada a integração dos modelos com maiores acurácias ao *software* de perícia computacional Autopsy, conforme descrito na Seção 3.4.3. As análises presentes nesta Seção focam na classificação de

Figura 4.17: Exemplos de imagens contendo possíveis drogas e corretamente identificadas pelo modelo.

imagens objeto de perícia contendo documentos gerais e documentos de identificação e no reconhecimento de armas em arquivos de imagem obtidos da *internet*.

O módulo de classificação desenvolvido para o Autopsy utilizou os modelos VGG16, obtendo acurácia de 97,9% na classificação de documentos e não documentos, conforme exposto na matriz de confusão (a) presente na Figura 4.18. Nesse teste realizado, dos arquivos contendo documentos de identificação, o módulo conseguiu classificar corretamente 78,8% dessas imagens, ao passo que 97,2% das imagens com documentos gerais foram acertadamente classificadas. A categorização realizada nessa última etapa de classificação pode ser analisada por meio da matriz de confusão (b) exibida na Figura 4.18.

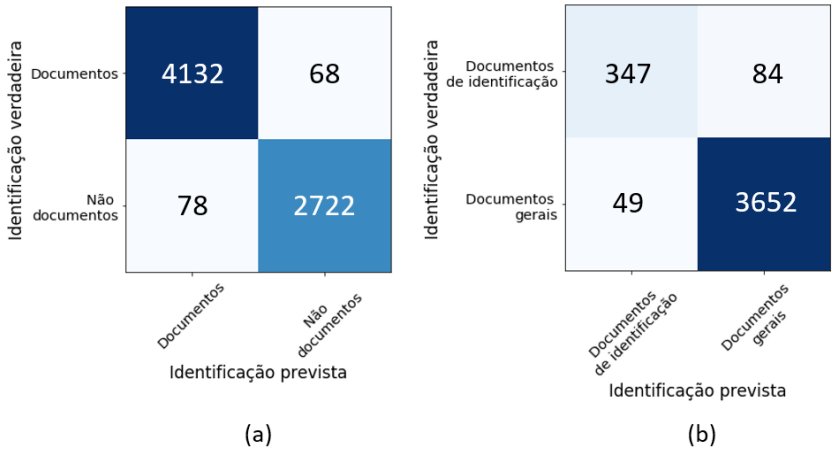



Figura 4.18: Matrizes de confusão para a classificação de imagens implantada no *software* Autopsy.

O módulo para detecção de armas conseguiu reconhecer esses objetos em 1930 imagens,

correspondendo a 77,2% dos arquivos contendo armas de fogo. Das 5000 imagens contendo objetos diversos, o detector identificou, equivocadamente, armas de fogo em 19,7% desses arquivos. O indicador *F1 score* final para o módulo de detecção de armas foi de 0,71. Salienta-se que, identificada uma arma de fogo na imagem, esse arquivo é classificado como "*com arma*", para fins de exibição no painel de arquivos do *software* Autopsy. Os resultados completos na detecção de armas estão expostos na Figura 4.19.

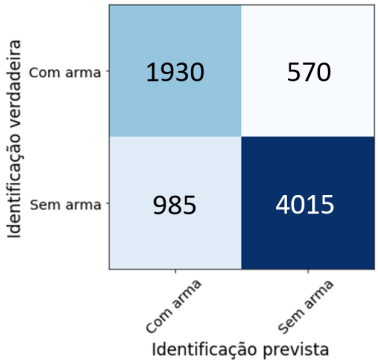



Figura 4.19: Matriz de confusão para as imagens analisadas após a detecção de armas.

A Figura 4.20 exibe *thumbnails* de imagens contendo armas identificadas pelo módulo desenvolvido. As imagens pertencentes à categoria de falsos-positivos foram analisadas e descobriram-se padrões relevantes. A maioria dos arquivos equivocadamente indicados como contendo armas de fogo possuíam, na verdade, aviões ou objetos em formato linear e pontiagudo, como luminárias, telefones, segmentos de madeira, punho de guidão de motocicleta e *skates*, por exemplo. Objetos sendo manuseados por pessoas, como celulares, raquetes de tênis e tacos de beisebol, por exemplo, também foram erroneamente reconhecidos como armas de fogo. Imagens do conjunto de falsos-positivos contendo objetos com alta probabilidade de serem armas, segundo o modelo, estão exemplificadas na Figura 4.21.

## 4.5 Critérios de reprodutibilidade

A etapa de reprodutibilidade deste projeto seguiu os itens descritos em *The Machine Learning Reproducibility Checklist* versão 2.0, proposta por Joelle Pineau [112] e disponível em [113]. O *checklist* apresenta critérios de reprodutibilidade que devem ser seguidos por projetos de aprendizado de máquina, estando dividido em cinco categorias.

Para todos os **modelos** e **algoritmos** apresentados, verificar se estão incluídos:

– *Uma descrição clara dos elementos matemáticos, algoritmos e/ou modelos*: as descrições de todos os algoritmos desenvolvidos estão disponíveis em [114].

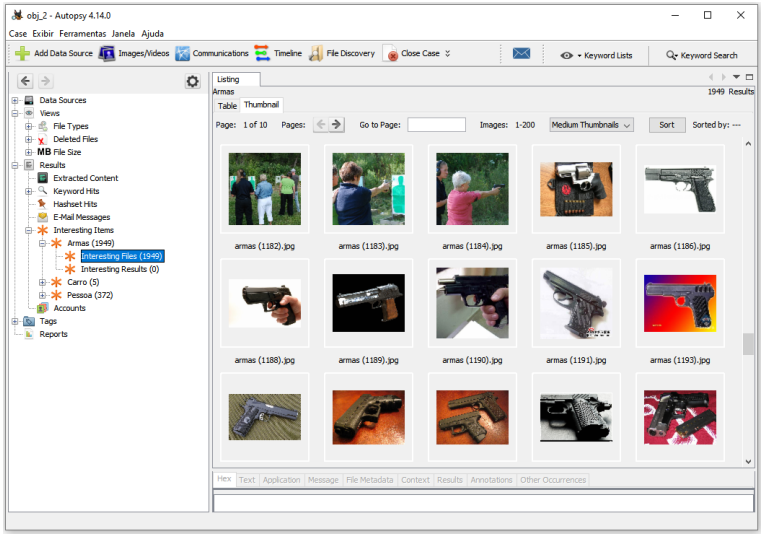



Figura 4.20: Tela do *software* Autopsy exibindo imagens categorizadas como contendo armas, após o processo de detecção.

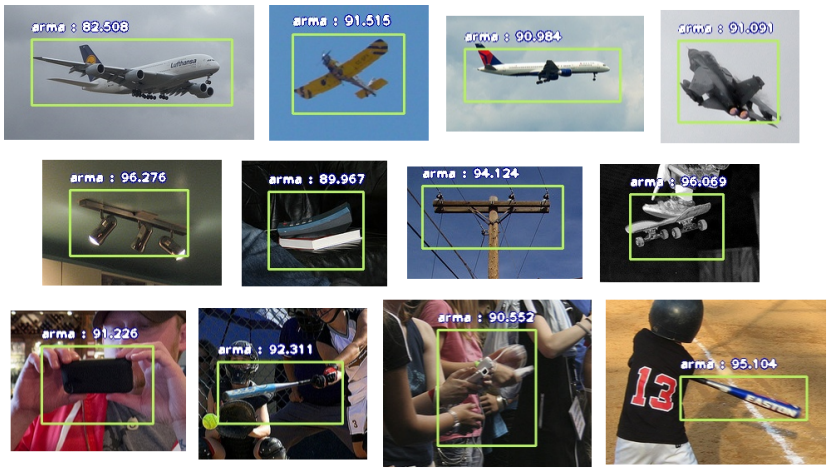



Figura 4.21: Exemplos de imagens contendo elementos erroneamente identificados como sendo armas de fogo.

– *Uma explicação clara de cada suposição*: não aplicável.
– *Análise de complexidade de cada algoritmo*: os algoritmos utilizados possuem ordem de complexidade de tempo e espaço $O(n)$, em que n representa o número de imagens.

Para cada **declaração teórica** verificar se estão incluídos:
– *Uma clara explicação da declaração*: não aplicável.
– *Uma prova completa de cada declaração*: não aplicável.

Para cada **conjunto de dados** verificar se estão incluídos:
– *As estatísticas relevantes, como número de exemplos*: vide Seções 3.3 e 4.4.

- *Detalhes da divisão dos conjuntos de treinamento, validação e teste*: vide Seções 3.4.1 e 3.4.2.
- *Uma explicação de cada dado que foi excluído, e toda a etapa de pré-processamento*: vide Seção 3.2.
- *Um link para download de uma versão do conjunto de dados ou ambiente de simulação*: em virtude da confidencialidade dos dados presentes nas imagens objeto de estudo, esses arquivos não poderão ser disponibilizados. Porém, as características das imagens utilizadas, a descrição do processo de obtenção e os endereços eletrônicos dos repositórios de imagens públicas extraídas da *internet* estão presentes nas Seções 3.3 e 4.4.
- *Para novos dados coletados, uma descrição completa do processo de coleta, como instruções para anotações e métodos para controle de qualidade*: vide Seções 3.2, 3.3 e 4.4.

Para todo **código** compartilhado relacionado com o projeto, verificar se estão incluídos:

- *Especificação das dependências*: vide Seções 3.4.1 e 3.4.2.
- *Código de treinamento*: disponível em [114].
- *Código de avaliação*: disponível em [114].
- *Modelos pré-treinados*: disponível em [114].
- *Arquivos README com tabelas dos resultados acompanhadas de comandos precisos para a execução, a fim de produzir esses resultados*: disponível em [114].

Para todo **resultado experimental** obtido, verificar se estão incluídos:

- *A variação de hiper parâmetros considerada, o método utilizado para selecionar a melhor configuração de hiper parâmetros e a especificação de todos os hiper-parâmetros utilizados na geração dos resultados*: vide Seções 3.4.1, 3.4.2 e 4.3.
- *O número exato de execuções de treinamento e validação*: vide Seções 3.4.1, 3.4.2, 4.1 e 4.3.
- *Uma clara definição das medidas ou estatísticas utilizadas nos resultados*: vide Seções 2.5.9, 4.1, 4.3 e 4.4.
- *Uma descrição dos resultados com medidas de tendência central e variação*: vide Seção 4.3.
- *O tempo de execução médio de cada execução ou estimação do custo de energia*: vide Seções 4.1 e 4.3.
- *Uma descrição da infraestrutura de computação usada*: vide Seção 3.4.

# Capítulo 5

# Conclusões e trabalhos futuros

Este projeto abordou a classificação de imagens e a detecção de objetos no âmbito da perícia criminal. Na etapa de classificação, a rede neural convolucional desenvolvida e as redes pré-treinadas utilizadas por meio de *transfer learning* apresentaram resultados positivos quanto à classificação de imagens com documentos de identificação, documentos gerais e não documentos. Apesar dos desempenhos semelhantes, a utilização de *transfer learning* por meio dos modelos pré-treinados VGG16 e VGG19 propiciou a geração de modelos específicos mais precisos para a tarefa de classificação proposta, oferendo acurácias de 98% e 99% para os classificadores A e B, respectivamente.

Após a etapa de classificação de imagens, ocorreu a fase de detecção de pessoas, carros e armas. Ao empregar uma estrutura hierárquica de classificação, o processo de detecção de objetos é acelerado ao filtrar previamente apenas as imagens com possíveis objetos de interesse, integrantes da categoria *não documentos*.

No que tange à detecção de armas, o treinamento utilizando o modelo pré-treinado YOLOv3 propiciou a geração de modelo específico com mAP de 0,85. A detecção de possíveis drogas apresentou mAP de 0,68, não gerando, portanto, modelos precisos para o reconhecimento desses elementos. Essa precisão inferior possivelmente está relacionada ao baixo quantitativo de imagens obtido para fins de treinamento, bem como à diversidade de formas com que a droga escolhida pode se apresentar.

Com o objetivo de auxiliar o perito criminal em suas análises, buscou-se integrar os modelos desenvolvidos neste trabalho ao *software* de perícia computacional Autopsy. O módulo, por meio do classificador A, obteve acurácia de 97,9% na classificação entre documentos e não documentos. Assim, ao final do processo de classificação, o módulo desenvolvido conseguiu classificar corretamente 78,8% das imagens contendo documentos de identificação e 97,2% das imagens contendo documentos gerais. No que tange à detecção de armas, o módulo criado foi capaz de reconhecer esse dispositivo em 77,2% das imagens fornecidas, apresentando *F1 score* de 0,71 para o processo de detecção. Além

dessas capacidades, foi possível introduzir a detecção automática de pessoas e carros ao *software* Autopsy por meio do modelo YOLOv3. Esses resultados demonstram o potencial do projeto desenvolvido ao propiciar um nível de precisão satisfatório nas tarefas de classificação de imagens e detecção de objetos no âmbito pericial.

A questão da reprodutibilidade também foi considerada ao longo do desenvolvimento deste trabalho. Para permitir a análise e a reprodução dos experimentos realizados, foram seguidas as etapas descritas no *checklist* apresentado, voltado para projetos de *machine learning*. Assim, instruções, algoritmos e modelos gerados foram disponibilizados a fim de facilitar a reprodução e a melhoria dos experimentos desenvolvidos.

Como etapas futuras deste projeto, destacam-se a melhoria da detecção de possíveis drogas (maconha), bem como a adição de outros objetos de interesse para fins de reconhecimento. Além da detecção de outros tipos de objetos e possíveis drogas, como cocaína, será realizada uma contínua coleta de imagens objetivando aumentar os conjuntos de treinamento para a geração de modelos mais precisos.

É esperado, também, o refinamento da etapa de classificação de imagens por meio da criação de subcategorias de documentos, como *screenshots* de aplicativos de mensageria, como *Whatsapp*, e contas de energia elétrica, por exemplo. Além do desenvolvimento do módulo destinado ao *software* Autopsy, está prevista a integração dos modelos de classificação de imagens e detecção de pessoas e objetos ao *software* IPED (Indexador e Processador de Evidências Digitais), desenvolvido pela Polícia Federal e utilizado por diversos institutos de criminalística do país.

# Referências


[1] *Constituição da República Federativa do Brasil*. 1988. http://www.planalto.gov.br/ccivil_03/constituicao/constituicao.htm. 1

[2] *Código de Processo Penal*. 1941. http://www.planalto.gov.br/ccivil_03/decreto-lei/del3689.htm. 1

[3] Eleutério, Pedro Monteiro e Marcio Pereira Machado: *Desvendando a Computação Forense*. 2011. 1

[4] McGuire, Mike e Samantha Dowling: *Cyber crime: A review of the evidence*. Relatório Técnico, 2013. https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/246749/horr75-summary.pdf. 2

[5] Meirelles, Fernando e @ Fgv Br: *29ª Pesquisa Anual do Uso de TI*. Relatório Técnico, 2018. www.fgv.br/cia/pesquisa. 2

[6] Morgan, Steve: *2017 Cybercrime Report*. Relatório Técnico, 2017. https://1c7fab3im83f5gqiow2qqs2k-wpengine.netdna-ssl.com/2015-wp/wp-content/uploads/2017/10/2017-Cybercrime-Report.pdf. 2

[7] Shalaginov, Andrii, Jan William Johnsen e Katrin Franke: *Cyber crime investigations in the era of big data*. Em *2017 IEEE International Conference on Big Data (Big Data)*, páginas 3672–3676. IEEE, dec 2017, ISBN 978-1-5386-2715-0. http://ieeexplore.ieee.org/document/8258362/. 3

[8] Zupic, Ivan e Tomaž Čater: *Bibliometric Methods in Management and Organization*. Organizational Research Methods, 18(3):429–472, jul 2015, ISSN 1094-4281. http://journals.sagepub.com/doi/10.1177/1094428114562629. 6

[9] Melo, Ari e Mariano Arimariano Br: *Revisão da Literatura: Apresentação de uma Abordagem Integradora*. ISBN 978-84-697-5592-1. 6, 7

[10] Vogel, Rick e Wolfgang H. Güttel: *The Dynamic Capability View in Strategic Management: A Bibliometric Review*. International Journal of Management Reviews, 15(4):n/a–n/a, nov 2012, ISSN 14608545. http://doi.wiley.com/10.1111/ijmr.12000. 7

[11] Arezoo, Aghaei Chadegani, Hadi Salehi, Md Yunus, Hadi Farhadi, Masood Fooladi, Maryam Farhadi, Nader Ale Ebrahim, Arezoo Aghaei Chadegani, Melor Md

Yunus, & Nader e Ale Ebrahim: *A Comparison between Two Main Academic Literature Collections: Web of Science and Scopus Databases*. https://hal.archives-ouvertes.fr/hal-00819821. 8

[12] Lowe, David G.: *Distinctive Image Features from Scale-Invariant Keypoints*. International Journal of Computer Vision, 60(2):91–110, ISSN 0920-5691. https://link.springer.com/article/10.1023/B:VISI.0000029664.99615.94. 10, 14

[13] Melgani, Farid, Lorenzo Bruzzone e Senior Member: *Classification of Hyperspectral Remote Sensing Images With Support Vector Machines*. IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, 42(8), 2004. https://rslab.disi.unitn.it/papers/R30-TGARS-SVM-HYPER.pdf. 10

[14] Krizhevsky, Alex, Ilya Sutskever e Geoffrey E Hinton: *ImageNet Classification with Deep Convolutional Neural Networks*. Relatório Técnico. http://code.google.com/p/cuda-convnet/. 11

[15] Li, Jingjing, Ke Lu, Zi Huang, Lei Zhu e Heng Tao Shen: *Transfer Independently Together: A Generalized Framework for Domain Adaptation*. IEEE Transactions on Cybernetics, 49(6):2144–2155, jun 2019, ISSN 21682267. 11

[16] Wang, Qi, Xiang He e Xuelong Li: *Locality and structure regularized low rank representation for hyperspectral image classification*. IEEE Transactions on Geoscience and Remote Sensing, 57(2):911–923, feb 2019, ISSN 01962892. 11

[17] LeCun, Yann, Yoshua Bengio e Geoffrey Hinton: *Deep learning*. Nature, 521(7553):436–444, may 2015, ISSN 0028-0836. http://www.nature.com/articles/nature14539. 14

[18] He, Kaiming, Xiangyu Zhang, Shaoqing Ren e Jian Sun: *Deep residual learning for image recognition*. Em *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-December, páginas 770–778. IEEE Computer Society, dec 2016, ISBN 9781467388504. 14

[19] *WordItOut – enjoy word clouds, create word art & gifts*. https://worditout.com/, acesso em 2020-05-09. 15

[20] Zhao, Zhong Qiu, Peng Zheng, Shou tao Xu e Xindong Wu: *Object Detection with Deep Learning: A Review*. jul 2018. http://arxiv.org/abs/1807.05511. 15, 49

[21] Ayzenberg, Vladislav e Stella F. Lourenco: *Skeletal descriptions of shape provide unique perceptual information for object recognition*. Scientific Reports, 9(1):1–13, dec 2019, ISSN 20452322. 15

[22] Amerini, I., L. Ballan, R. Caldelli, A. Del Bimbo e G. Serra: *A SIFT-Based Forensic Method for Copy–Move Attack Detection and Transformation Recovery*. IEEE Transactions on Information Forensics and Security, 6(3):1099–1110, sep 2011, ISSN 1556-6013. http://ieeexplore.ieee.org/document/5734842/. 16

[23] Huang, Yanping, Wei Lu, Wei Sun e Dongyang Long: *Improved DCT-based detection of copy-move forgery in images*. Forensic Science International, 206(1-3):178–184, mar 2011, ISSN 03790738. 16

[24] Chen, Xu, Jianjun Li, Yanchao Zhang, Yu Lu e Shaoyu Liu: *Automatic feature extraction in X-ray image based on deep learning approach for determination of bone age*. Future Generation Computer Systems, oct 2019, ISSN 0167739X. 19

[25] Ortiz, AG, C. Costa, RHA Silva, MGH Biazevic e E. Michel-Crosato: *Sex estimation: Anatomical references on panoramic radiographs using Machine Learning*. Forensic Imaging, 20:200356, mar 2020, ISSN 26662256. 19

[26] Liu, Ying, Yanan Peng, Daxiang Li, Jiulun Fan e Yun Li: *Crime scene investigation image retrieval with fusion CNN features based on transfer learning*. Em *ACM International Conference Proceeding Series*, páginas 68–72, New York, New York, USA, mar 2018. Association for Computing Machinery, ISBN 9781450364683. `http://dl.acm.org/citation.cfm?doid=3195588.3195605`. 19, 20

[27] Giverts, Pavel, Saad Sofer, Yosef Solewicz e Boris Varer: *Firearms identification by the acoustic signals of their mechanisms*. Forensic Science International, 306:110099, jan 2020, ISSN 18726283. 19

[28] Tsai, Min Jen, Yu Han Tao e Imam Yuadi: *Deep learning for printed document source identification*. Signal Processing: Image Communication, 70:184–198, feb 2019, ISSN 09235965. 20

[29] Rehman, Arshia, Saeeda Naz e Muhammad Imran Razzak: *Writer identification using machine learning approaches: a comprehensive review*. Multimedia Tools and Applications, 78(8):10889–10931, apr 2019, ISSN 15737721. 20

[30] Jung, Jaemin, Jongmoo Choi, Seong Je Cho, Sangchul Han, Minkyu Park e Youngsup Hwang: *Android malware detection using convolutional neural networks and data section images*. Em *Proceedings of the 2018 Research in Adaptive and Convergent Systems, RACS 2018*, páginas 149–153, New York, New York, USA, oct 2018. Association for Computing Machinery, Inc, ISBN 9781450358859. `http://dl.acm.org/citation.cfm?doid=3264746.3264780`. 20

[31] Hendler, Danny, Shay Kels e Amir Rubin: *Detecting malicious powershell commands using deep neural networks*. Em *ASIACCS 2018 - Proceedings of the 2018 ACM Asia Conference on Computer and Communications Security*, páginas 187–197, New York, New York, USA, may 2018. Association for Computing Machinery, Inc, ISBN 9781450355766. `http://dl.acm.org/citation.cfm?doid=3196494.3196511`. 20

[32] Ali, Muhammad, Stavros Shiaeles, Nathan Clarke e Dimitrios Kontogeorgis: *A proactive malicious software identification approach for digital forensic examiners*. Journal of Information Security and Applications, 47:139–155, aug 2019, ISSN 22142126. 20

[33] Wang, Yanchao, Zhongqian Su e Dayi Song: *File fragment type identification with convolutional neural networks*. Em *ACM International Conference Proceeding Series*, páginas 41–47, New York, New York, USA, may 2018. Association for Computing Machinery, ISBN 9781450364324. http://dl.acm.org/citation.cfm?doid=3231884.3231889. 20

[34] Mohamed, Learning Alji e Chougdali Khalid: *Detection of timestamps tampering in ntfs using machine*. Em *Procedia Computer Science*, volume 160, páginas 778–784. Elsevier B.V., jan 2019. 20

[35] Song, Jinho, Chae Ho Cho e Yoojae Won: *Analysis of operating system identification via fingerprinting and machine learning*. Computers and Electrical Engineering, 78:1–10, sep 2019, ISSN 00457906. 20

[36] Zou, Ying, Ge Zhang e Leian Liu: *Research on image steganography analysis based on deep learning*. Journal of Visual Communication and Image Representation, 60:266–275, apr 2019, ISSN 10959076. 20

[37] Jung, Ki Hyun: *A study on machine learning for steganalysis*. Em *ACM International Conference Proceeding Series*, páginas 12–15. Association for Computing Machinery, jan 2019, ISBN 9781450366120. 20

[38] Lin, Yuzhen, Rangding Wang, Diqun Yan, Li Dong e Xueyuan Zhang: *Audio steganalysis with improved convolutional neural network*. Em *IH and MMSec 2019 - Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*, páginas 210–215, New York, New York, USA, jul 2019. Association for Computing Machinery, Inc, ISBN 9781450368216. http://dl.acm.org/citation.cfm?doid=3335203.3335736. 20

[39] Mo, Huaxiao, Bolin Chen e Weiqi Luo: *Fake faces identification via convolutional neural network*. Em *IH and MMSec 2018 - Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*, páginas 43–47, New York, New York, USA, jun 2018. Association for Computing Machinery, Inc, ISBN 9781450356251. http://dl.acm.org/citation.cfm?doid=3206004.3206009. 20

[40] Liu, Huajian, Martin Steinebach e Kathrin Schölei: *Improved manipulation detection with convolutional neural network for JPEG images*. Em *ACM International Conference Proceeding Series*, páginas 1–6, New York, New York, USA, aug 2019. Association for Computing Machinery, ISBN 9781450371643. http://dl.acm.org/citation.cfm?doid=3339252.3340526. 20

[41] Athanasiadou, Eleni, Zeno Geradts e Erwin Van Eijk: *Camera recognition with deep learning*. Forensic Sciences Research, 3(3):210–218, jul 2018, ISSN 2096-1790. https://www.tandfonline.com/doi/full/10.1080/20961790.2018.1485198. 20

[42] Yang, Pengpeng, Daniele Baracchi, Rongrong Ni, Yao Zhao, Fabrizio Argenti e Alessandro Piva: *A Survey of Deep Learning-Based Source Image Forensics*. Journal of Imaging, 6(3):9, mar 2020, ISSN 2313-433X. https://www.mdpi.com/2313-433X/6/3/9. 20

[43] Bedeli, Marianna, Zeno Geradts e Erwin van Eijk: *Clothing identification via deep learning: forensic applications*. Forensic Sciences Research, 3(3):219–229, jul 2018, ISSN 24711411. 20

[44] Domingues, Patricio e Alexandre Frazão Rosário: *Deep learning-based facial detection and recognition in still images for digital forensics*. Em *ACM International Conference Proceeding Series*, páginas 1–10, New York, New York, USA, aug 2019. Association for Computing Machinery, ISBN 9781450371643. `http://dl.acm.org/citation.cfm?doid=3339252.3340107`. 20, 54

[45] Silva, Pedro Monteiro da e Mateus de Castro Polastro: *An overview of NuDetective Forensic Tool and its usage to combat child pornography in Brazil*. 2014. `http://sedici.unlp.edu.ar/handle/10915/42067`. 21

[46] George, Raburu, Omollo Richard e Okumu Daniel: *International Journal of Computer Science and Mobile Computing Applying Data Mining Principles in the Extraction of Digital Evidence*. Relatório Técnico 3, 2018. `www.ijcsmc.com`. 21

[47] Lisboa Perez, Mauricio, Vanessa Testoni e Anderson Rocha: *Video pornography detection through deep learning techniques and motion information*. Relatório Técnico. `http://sibgrapi2017.ic.uff.br/e-proceedings/assets/papers/WTD/WD1.pdf`. 21

[48] Marturana, Fabio e Simone Tacconi: *A Machine Learning-based Triage methodology for automated categorization of digital media*. Digital Investigation, 10(2):193–204, sep 2013, ISSN 1742-2876. `https://www.sciencedirect.com/science/article/pii/S1742287613000029`. 21

[49] Saikia, Surajit, E. Fidalgo, Enrique Alegre e Laura Fernández-Robles: *Object Detection for Crime Scene Evidence Analysis Using Deep Learning*. páginas 14–24. Springer, Cham, 2017. `http://link.springer.com/10.1007/978-3-319-68548-9_2`. 21

[50] Mayer, Felix e Martin Steinebach: *Forensic Image Inspection Assisted by Deep Learning*. Em *Proceedings of the 12th International Conference on Availability, Reliability and Security - ARES '17*, páginas 1–9, New York, New York, USA, 2017. ACM Press, ISBN 9781450352574. `http://dl.acm.org/citation.cfm?doid=3098954.3104051`. 22

[51] Nassif, Luís Filipe da Cruz e Eduardo Raul Hruschka: *Document Clustering for Forensic Analysis: An Approach for Improving Computer Inspection*. IEEE Transactions on Information Forensics and Security, 8(1):46–54, jan 2013, ISSN 1556-6013. `http://ieeexplore.ieee.org/document/6327658/`. 22

[52] Dåderman, Antonia, Sara Rosander, Kth Skolan, För Elektroteknik e Och Datavetenskap: *Evaluating Frameworks for Implementing Machine Learning in Signal Processing A Comparative Study of CRISP-DM, SEMMA and KDD*. Relatório Técnico. `http://www.diva-portal.se/smash/get/diva2:1250897/FULLTEXT01.pdf`. 23

[53] Han, Jiawei. e Micheline. Kamber: *Data mining : concepts and techniques*. Elsevier, 2006, ISBN 9780080475585. 23

[54] Wirth, Rüdiger e Rüdiger Wirth: *CRISP-DM: Towards a standard process model for data mining*. PROCEEDINGS OF THE FOURTH INTERNATIONAL CONFERENCE ON THE PRACTICAL APPLICATION OF KNOWLEDGE DISCOVERY AND DATA MINING, páginas 29–39, 2000. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.198.5133. 23

[55] Gonzalez, Rafael C. e Richard E. Woodz: *Processamento Digital De Imagens*. Pearson, São Paulo, 3ª edição, 2011, ISBN 9788576054016. 24, 25

[56] Jahne, Bernd: *Digital Image Processing*. Springer Berlin Heidelberg, 5ª edição, 2002. 24

[57] Martins, Samuel Botter: *Introdução ao Processamento Digital de Imagens*. Relatório Técnico, UNICAMP. https://www.ic.unicamp.br/~ra144681/misc/files/ApostilaProcDeImagesParteI.pdf. 25

[58] Burger, Wilhelm e Mark J. Burge: *Digital Image Processing - An Algorithmic Introduction Using Java*. 2016. 26, 27

[59] Miano, John: *Compressed image file formats : JPEG, PNG, GIF, XBM, BMP*. Addison Wesley, 1999, ISBN 9780201604436. 27

[60] Murty, M. N. (Maddipati Narasimha) e V. Susheela. Devi: *Pattern recognition : an algorithmic approach*. Springer, 2011, ISBN 9780857294951. 27

[61] Beyerer, Jürgen, Matthias Richter e Matthias Nagel: *Pattern Recognition*. De Gruyter, Berlin, Boston, dec 2017, ISBN 9783110537949. http://www.degruyter.com/view/books/9783110537949/9783110537949/9783110537949.xml. 27

[62] Conci, Aura, Eduardo Azevedo e Fabiana Rodrigues Leta: *Computacao Grafica. Volume 2*. ISBN 9788535223293. https://www.sciencedirect.com/book/9788535223293/computacao-grafica-volume-2. 27

[63] HAYKIN, S S: *Redes Neurais - 2ed.* Bookman, 2001, ISBN 9788573077186. https://books.google.com.br/books?id=lBp0X5qfyjUC. 27, 29, 30, 31, 32

[64] Zerdoumi, Saber, Aznul Qalid Md Sabri, Amirrudin Kamsin, Ibrahim Abaker Targio Hashem, Abdullah Gani, Saqib Hakak, Mohammed Ali Al-garadi e Victor Chang: *Image pattern recognition in big data: taxonomy and open challenges: survey*. Multimedia Tools and Applications, 77(8):10091–10121, apr 2018, ISSN 1380-7501. http://link.springer.com/10.1007/s11042-017-5045-7. 28

[65] Rosenfeld, A.: *Image pattern recognition*. Proceedings of the IEEE, 69(5):596–605, 1981, ISSN 0018-9219. http://ieeexplore.ieee.org/document/1456295/. 28

[66] Hall, E.L., R.P. Kruger, S.J. Dwyer, D.L. Hall, R.W. Mclaren e G.S. Lodwick: *A Survey of Preprocessing and Feature Extraction Techniques for Radiographic Images*. IEEE Transactions on Computers, C-20(9):1032–1044, sep 1971, ISSN 0018-9340. http://ieeexplore.ieee.org/document/1671992/. 28

[67] Erpen, Luís Renato Cruz: *Reconhecimento de Padrões em Imagens por Descritores de Forma*. Relatório Técnico, Porto Alegre, 2004. https://www.lume.ufrgs.br/bitstream/handle/10183/27662/000766057.pdf?sequence=1. 28

[68] Zhou, Shuai, Yanyan Liang, Jun Wan e Stan Z. Li: *Facial Expression Recognition Based on Multi-scale CNNs*. páginas 503–510. Springer, Cham, oct 2016. http://link.springer.com/10.1007/978-3-319-46654-5_55. 28

[69] Braga, Antonio de Padua., Andre Carlos Ponce de Leon Ferreira. Carvalho e Teresa Bernarda. Ludermir: *Redes neurais artificiais : teoria e aplicacoes*. LTC Editora, 2007, ISBN 8521615647. 29, 30

[70] Ebermam, Elivelto e Renato Krohling: *Uma Introdução Compreensiva às Redes Neurais Convolucionais: Um Estudo de Caso para Reconhecimento de Caracteres Alfabéticos*. Revista de Sistemas de Informação da FSMA, 22:49–59, 2018. http://www.fsma.edu.br/si/sistemas.html. 29, 33, 34

[71] Loy, James: *Neural network projects with Python : the ultimate guide to using Python to explore the true power of neural networks through six projects*. 2019, ISBN 9781789138900. 31

[72] Aggarwal, Charu C.: *Neural Networks and Deep Learning*. Springer International Publishing, 2018. 33, 37, 38, 39, 40, 41, 46

[73] Hubel, D. H. e T. N. Wiesel: *Receptive fields of single neurones in the cat's striate cortex*. The Journal of Physiology, 148(3):574–591, oct 1959, ISSN 14697793. 33

[74] Ponti, Moacir A e Gabriel B Paranhos Da Costa: *Capítulo 3 Como funciona o Deep Learning*. ISBN 9788576694007. http://conteudo.icmc.usp.br/pessoas/moacir/papers/Ponti_Costa_Como-funciona-o-Deep-Learning_2017.pdf. 34

[75] Gomes Vargas, Ana Caroline, Aline Marins, Paes Carvalho e Cristina Nader Vasconcelos: *Um estudo sobre redes neurais convolucionais e sua aplicação em detecção de pedestres*. Em *Proceedings of the XXIX Conference on Graphics, Patterns and Images*, páginas 1–4, 2016. http://gibis.unifesp.br/sibgrapi16/eproceedings/wuw/7.pdf. 34, 35

[76] Khan, Salman, Hossein Rahmani, Syed Afaq Ali Shah e Mohammed Bennamoun: *A Guide to Convolutional Neural Networks for Computer Vision*. Synthesis Lectures on Computer Vision, 8(1):1–207, feb 2018, ISSN 2153-1056. 35, 36, 37, 38, 39, 41, 47, 48

[77] Agarwal, Shivang, Jean Ogier Du Terrail e Frédéric Jurie: *Recent Advances in Object Detection in the Age of Deep Convolutional Neural Networks*. sep 2018. http://arxiv.org/abs/1809.03193. 35, 36, 46

[78] Simonyan, Karen e Andrew Zisserman: *Very Deep Convolutional Networks for Large-Scale Image Recognition*. sep 2014. http://arxiv.org/abs/1409.1556. 37

[79] Rezende, Edmar, Guilherme Ruppert, Tiago Carvalho, Antonio Theophilo, Fabio Ramos e Paulo de Geus: *Malicious Software Classification Using VGG16 Deep Neural Network's Bottleneck Features*. Em *Advances in Intelligent Systems and Computing*, volume 738, páginas 51–59. Springer Verlag, 2018, ISBN 9783319770277. 37

[80] Mahdianpari, Masoud, Bahram Salehi, Mohammad Rezaee, Fariba Mohammadimanesh e Yun Zhang: *Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery*. Remote Sensing, 10(7):1119, jul 2018, ISSN 2072-4292. `http://www.mdpi.com/2072-4292/10/7/1119`. 37

[81] Villán, Alberto F: *Mastering OpenCV 4 with Python*. 2019, ISBN 9781789344912. 38

[82] Chollet, François: *Xception: Deep learning with depthwise separable convolutions*. Em *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-January, páginas 1800–1807. Institute of Electrical and Electronics Engineers Inc., nov 2017, ISBN 9781538604571. 39, 40

[83] He, Kaiming, Xiangyu Zhang, Shaoqing Ren e Jian Sun: *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*. feb 2015. `http://arxiv.org/abs/1502.01852`. 40

[84] Pan, Sinno Jialin e Qiang Yang: *A Survey on Transfer Learning*. 2009. `http://socrates.acadiau.ca/courses/comp/dsilver/NIPS95`. 41, 42

[85] Michelucci, Umberto: *Advanced Applied Deep Learning*. Apress, 2019. 42

[86] Weiss, Karl, Taghi M. Khoshgoftaar e Ding Ding Wang: *A survey of transfer learning*. Journal of Big Data, 3(1):9, dec 2016, ISSN 21961115. `http://journalofbigdata.springeropen.com/articles/10.1186/s40537-016-0043-6`. 42

[87] Dipanjan, Sarkar, Raghav Bali e Tamoghna Ghosh: *Hands-On Transfer Learning with Python*. Packt Publishing, 2018, ISBN 9781788831307. 43

[88] Nisbet, Robert., John F. (John Fletcher) Elder e Gary. Miner: *Handbook of statistical analysis and data mining applications*. Academic Press/Elsevier, 2009, ISBN 9780080912035. 43

[89] De Castro, L.N.: *Introdução à Mineração de Dados. Conceitos Básicos, Algoritmos e Aplicações*. 2016. 43

[90] Berka, Petr, Jan Rauch e Djamel Abdelkader Zighed (editores): *Data Mining and Medical Knowledge Management*. IGI Global, 2009, ISBN 9781605662183. `http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-60566-218-3`. 44

[91] Galdino dos Santos, Miqueas; Ribeiro Lins Júnior, Paulo.: *Avaliação de Técnicas Estatísticas para Detecção de Anomalias Aplicadas à Internet das Coisas*. Revista de Tecnologia da Informação e Comunicação, 8(1):13–18, may 2018. http://rtic.com.br/index.php/rtic/article/view/95. 44

[92] Rezatofighi, Hamid, Nathan Tsoi, Junyoung Gwak, Amir Sadeghian, Ian Reid e Silvio Savarese: *Generalized intersection over union: A metric and a loss for bounding box regression*. Em *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, páginas 658–666. IEEE Computer Society, jun 2019, ISBN 9781728132938. 44

[93] Grauman, Kristen e Bastian Leibe: *Visual object recognition*, volume 11. Morgan & Claypool Publishers, apr 2011, ISBN 9781598299687. 46, 77

[94] Hassaballah, Mahmoud e Ali Ismail Awad: *Deep learning in computer vision : principles and applications*. CRC Press, 1ª edição, 2020, ISBN 1138544426. 46, 47, 50, 51

[95] Girshick, Ross: *Fast R-CNN*. apr 2015. http://arxiv.org/abs/1504.08083. 47

[96] Pfeifer, Lienhard.: *Pedestrian Detection Algorithms Using Shearlets*. Logos Verlag Berlin, jan 2019, ISBN 9783832590130. 48, 49

[97] Girshick, Ross, Jeff Donahue, Trevor Darrell e Jitendra Malik: *Region-Based Convolutional Networks for Accurate Object Detection and Segmentation*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 38(1):142–158, jan 2016, ISSN 01628828. 48

[98] Ren, Shaoqing, Kaiming He, Ross Girshick e Jian Sun: *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6):1137–1149, jun 2017, ISSN 0162-8828. http://ieeexplore.ieee.org/document/7485869/. 48, 49

[99] Redmon, Joseph, Santosh Divvala, Ross Girshick e Ali Farhadi: *You Only Look Once: Unified, Real-Time Object Detection*. jun 2015. http://arxiv.org/abs/1506.02640. 49, 50, 51

[100] Reyes, Esteban, Cristopher GÃmez, Esteban Norambuena e Javier Ruiz-del Solar: *Near Real-Time Object Recognition for Pepper Based on Deep Neural Networks Running on a Backpack*. Em *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11374 LNAI, páginas 287–298. Springer Verlag, jun 2019, ISBN 9783030275433. 50, 51

[101] Zhao, Xia, Yingting Ni e Haihang Jia: *Modified object detection method based on YOLO*. Em *Communications in Computer and Information Science*, volume 773, páginas 233–244. Springer Verlag, oct 2017, ISBN 9789811073045. 50

[102] Ivie, Peter e Douglas Thain: *Reproducibility in scientific computing*. ACM Computing Surveys, 51(3):1–36, apr 2018, ISSN 15577341. https://dl.acm.org/doi/10.1145/3186266. 52

[103] Baker, Monya e Dan Penny: *Is there a reproducibility crisis?* Relatório Técnico 7604, may 2016. 52

[104] *Autopsy*. https://www.sleuthkit.org/autopsy/, acesso em 2020-05-06. 53

[105] Kleiman, Dave: *Official CHFI Study Guide (Exam 312-49) : for Computer Hacking Forensic Investigator*. Elsevier, Burlington, MA, USA, 2007, ISBN 9780080555713. 53

[106] *GitHub - tzutalin/labelImg*. https://github.com/tzutalin/labelImg, acesso em 2020-04-29. 58

[107] Everingham, Mark, Luc Van Gool, Christopher K. I. Williams, John Winn e Andrew Zisserman: *The Pascal Visual Object Classes (VOC) Challenge*. International Journal of Computer Vision, 88(2):303–338, jun 2010, ISSN 0920-5691. http://link.springer.com/10.1007/s11263-009-0275-4. 59

[108] *COCO - Common Objects in Context*. http://cocodataset.org/#download, acesso em 2020-06-14. 59, 60

[109] *How to train YOLOv3 on Google COLAB to detect custom objects (e.g: Gun detection)*. shorturl.at/eMUXY, acesso em 2020-06-14. 60

[110] *GitHub - OlafenwaMoses/ImageAI*. https://github.com/OlafenwaMoses/ImageAI, acesso em 2020-04-30. 64, 67

[111] Chollet, François: *Deep Learning with Phyton*. Manning, 2018, ISBN 9781617294433. 73, 74, 75

[112] Pineau, Joelle, Philippe Vincent-Lamarre, Koustuv Sinha e Vincent Larivière: *Improving Reproducibility in Machine Learning Research (A Report from the NeurIPS 2019 Reproducibility Program)* . 2020. 81

[113] *The Machine Learning Reproducibility Checklist*, apr 2020. https://www.cs.mcgill.ca/~jpineau/ReproducibilityChecklist.pdf, acesso em 2020-05-19. 81

[114] Almeida, Juliano R.: *Projeto ppca*. https://github.com/julianoppca/Projeto_PPCA_2020, 2020. 81, 83