UNIVERSIDADE FEDERAL DO PARANÁ

GUILHERME DE MORAES RESTANI

IDENTIFICAÇÃO DE FALTAS EM SISTEMAS DE DISTRIBUIÇÃO DE ENERGIA ELÉTRICA COM APRENDIZADO PROFUNDO E PROCESSAMENTO DE IMAGENS

CURITIBA

2019

GUILHERME DE MORAES RESTANI

IDENTIFICAÇÃO DE FALTAS EM SISTEMAS DE DISTRIBUIÇÃO DE ENERGIA ELÉTRICA COM APRENDIZADO PROFUNDO E PROCESSAMENTO DE IMAGENS

Trabalho apresentado à banca examinadora da Universidade Federal do Paraná, como requisito para a obtenção do título de bacharel em Engenharia Elétrica, sob a orientação do Prof. Dr. Leandro dos Santos Coelho.

CURITIBA

2019

AGRADECIMENTOS

Agradeço a meus pais e familiares, os que deram apoio às minha decisões e escolhas de carreira, além de todo o suporte para que este sonho se concretizasse.

Agradeço a Marcel E. Bocalão, pelo apoio ao nos momentos mais difíceis desta jornada. Sem você nada disso teria sido possível.

Agradeço a José F. Bianchi Filho, meu orientador de estágio, pelo aprendizado proporcionado no campo da inteligência artificial e pela amizade.

Agradeço a também a todos os amigos e colegas que estiveram comigo ao longo destes anos.

Por fim, agradeço a todos que se dedicaram ou se dedicam pelo desenvolvimento da ciência.

”If a machine is expected to be infallible,
it cannot also be intelligent” – Alan Turing

RESUMO

Falhas nos sistemas de distribuição de energia elétrica estão entre os fatores que mais impactam na qualidade e continuidade do fornecimento de energia elétrica, gerando custos e insatisfação dos usuários. Sendo assim, este projeto utiliza técnicas de aprendizado profundo e processamento digital de imagens para o desenvolvimento de uma aplicação capaz de detectar a presença de postes em imagens, bem como suas orientações, além da presença de chaves e seus estados (aberto ou fechado), culminando em um sistema que pode ser utilizado para detectar ou prever falhas em linhas de distribuição de energia elétrica. Através da pesquisa inicial, chegou-se a hipótese de que era possível de se utilizar de métodos que apresentavam bons resultados no banco de dados de imagens aéreas DOTA para a tarefa. Foi analisado o estado da arte e escolhida a rede SCRDet para implementação. Foi construído um banco de dados com imagens de postes e chaves, o qual veio a ser ampliado e ajustado posteriormente. Através das etapas de treinos, avaliação dos resultados e ajustes, foi possível chegar a um resultado satisfatório que comprova a viabilidade da aplicação.

Palavras-chaves: Poste. Chave Fusível. Aprendizado Profundo. Falta. Distribuição de Energia.

ABSTRACT

Faults in power distribution systems are among the factors that affect the most the quality and continuity of the electric power supply, leading to higher operational cost and user dissatisfaction. Thus, this project takes advantage of deep learning and digital image processing techniques for the development of an application capable of detecting the presence of poles in images, as well as their angles, and the presence of fuse cutouts and their status (open or closed), resulting in a system that can be used to detect or prevent faults in power distribution systems. Through the initial research, it was hypothesized that it is possible to use methods that perform well in DOTA aerial image database to accomplish this task. The state of the art in object detection was studied and the SCRDet network was chosen for implementation. A dataset containing images of poles and cutout fuses was developed later expanded and adjusted. Through the stages of training, results evaluation and adjustments, it was possible to reach a satisfactory result that has proven the viability of the application.

Key-words: Utility pole. Fuse Cutout. Deep Learning. Fault. Distribution Systems.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E DE SIGLAS

ANEEL Agência Nacional de Energia Elétrica

ANN Artificial Neural Network

AP Average Precision

CNN Convolutional Neural Network

DNN Deep Neural Network

FN Falsos Negativos

FP Falsos Positivos

HOG Histograma de Gradiente Orientado

IEEE Institute of Electrical and Electronics Engineers

IoU Intersection over Union

MLP Multilayer Perceptron

MNIST Modified National Institute of Standards and Technology

RPN Region Proposal Network

SVM Support Vector Machine

VN Verdadeiros Negativos

VP Verdadeiros Positivos

mAP mean Average Precision

LISTA DE SÍMBOLOS

x_j	sinal de entrada aplicado a uma sinapse j
j	uma sinapse qualquer
k	um neurônio qualquer
w_{kj}	peso de uma sinapse j conectada a um neurônio k
u_k	somatório das entradas de uma sinapse ponderadas pelo peso das conexões
b_k	bias
$\varphi(.)$	função de ativação de um neurônio k
v_k	saída de um neurônio k
$C(w, b)$	função de custo
w	conjunto de todos os pesos de uma rede
b	conjunto de todos os limiares de uma rede

SUMÁRIO

1 INTRODUÇÃO

A energia elétrica é essencial para a manutenção das atividades humanas, bem como para o desenvolvimento da sociedade. Além disso, a eletricidade, produzida e entregue através de sistemas de geração, transmissão e distribuição, é um dos maiores mercados consumidores do mundo (BROWN, 2008).

No Brasil, o setor de energia elétrica é dividido entre: geradoras, que produzem a energia; transmissoras, que realizam o transporte da energia produzida pelas geradoras para os centros consumidores; distribuidoras, que entregam a energia para a população; e comercializadoras, que são empresas com autorização para comercializar energia com consumidores livres, como grandes indústrias com grande necessidade de eletricidade, por exemplo.

A partir de 1997 iniciou-se no Brasil o processo de privatização na área de distribuição de energia elétrica, introduzindo novas empresas no mercado. Desta forma, as empresas de energia elétrica possuem a necessidade de reduzir custos com o objetivo de gerar lucro e se manterem competitivas. Contudo, a redução dos custos frequentemente causa impacto na qualidade do serviço oferecido.

Para proteger os clientes e garantir a qualidade do serviço existe a Agência Nacional de Energia Elétrica - ANEEL , uma autarquia vinculada ao Ministério de Minas e Energia que atua como a reguladora do setor. Entre as responsabilidades da ANEEL estão estabelecer índices de desempenho para os sistemas, além de fiscalizar a atuação das empresas. Além disso, cabe a ela aplicar multas às companhias em caso de descumprimento de suas regras.

Como o sistema de distribuição é responsável por cerca de 90% dos problemas de confiabilidade dos consumidores (BROWN, 2008), este é um dos tópicos mais importantes no escopo desta indústria, já que este é um fator que irá impactar diretamente no custo da eletricidade e na satisfação do consumidor.

Sendo assim, torna-se imprescindível que as companhias encontrem maneiras eficazes de reduzir seus custos sem que isto culmine na perda da qualidade do serviço prestado, além de encontrar novas formas de garantir o fornecimento de energia aos seus consumidores.

Nos últimos anos, uma das área da ciência que apresentaram grande desenvolvimento é a da inteligencia artificial, principalmente sua sub-área do aprendizado profundo (deep learning), que diz respeito ao desenvolvimento e implementação de algoritmos de redes neurais artificiais (do inglês *Artificial Neural Networks* - ANNs). Estas, por sua vez, são estruturas computacionais inspiradas pelo funcionamento das

redes neurais dos seres vivos. Assim como o cérebro humano, a rede aprende e faz descobertas através de bons exemplos e de seus erros e correções, se tornando capaz de adquirir conhecimento através da experiência. Desta forma, o modelo computacional pode ser capaz de executar tarefas de classificação diretamente a partir de imagens, textos ou sons. Modelos de aprendizado profundo tem atingido uma precisão de estado da arte e por isso tem sido muito aplicados em diversas áreas, como é o caso da área de visão computacional, uma área da ciência e tecnologia que diz respeito ao desenvolvimento de sistemas artificiais capazes de extrair conhecimento a partir de imagens. Isto ocorreu principalmente em função da redução dos custos das placas gráficas e das câmeras, além da quantidade de dados em abundância.

Desta forma, este projeto irá investigar e se utilizar dos avanços mais recentes da inteligência artificial para o desenvolvimendo de uma aplicação que colabore para a qualidade e confiabilidade do sistema de distribuição de energia elétrica.

1.1 PROBLEMA

Para serem capazes de cumprir as regras impostas pela ANEEL e oferecer um serviço de qualidade, as concessionárias se voltaram para a avaliação preditiva de seus sistemas, ou seja, observando pontos passíveis de falha e avaliando estratégias para contorná-los (GUIMARÃES, 2006).

Vários fatores podem impactar na qualidade e continuidade da distribuição de energia elétrica. Podem ser falhas em equipamentos oriundas de problemas de fabricação, danos durante a entrega, instalação incorreta, envelhecimento e condições de operação. Podem ser causadas também por animais ou vegetação, condições climáticas severas e até mesmo erro humano (BROWN, 2008).

Um dos problemas que podem ser detectados ou até mesmo previstos é o da queda de postes. Através da observação da sua inclinação é possível realizar um prognostico sobre sua condição.

Nos últimos anos a operação dos sistemas de energia elétrica está cada vez mais automatizada, suprimindo alguns tipos de falhas e fazendo com que novas possam surgir. Um tipo de falha que pode ocorrer da automatização é o da configuração das chaves do tipo fusível ser diferente daquela ordenada pela sistema de controle. Por exemplo, deseja-se abrir uma chave, porém suas condições de envelhecimento em função do clima faz com que ela permaneça fechada.

Sendo assim, outro problema que pode ser detectado com a utilização de visão computacional é a verificação da configuração das chave fusível.

Estes dois problemas em específicos serão endereçados pois eles permitirão observar se os sistemas de aprendizado profundo são capazes de detectar objetos

que compõem uma rede de distribuição, como os postes e as chaves fusível, além de mostrar se é possível um conhecimento da imagens que seja útil para a melhoria do sistema.

1.2 OBJETIVOS

Nesta seção serão apresentados os objetivos gerais e específicos do projeto.

1.2.1 Objetivo Geral

Investigar e implementar técnicas de aprendizado profundo no escopo da engenharia elétrica através da aplicação de métodos de reconhecimento de padrões e objetos em imagens, visando o desenvolvimento de uma aplicação que auxilie na identificação do tipo e do local da falta em sistemas de distribuição de energia elétrica.

1.2.2 Objetivos Específicos

Os seguintes são os objetivos específicos do projeto:

- analisar e implementar conceitos de inteligência artificial e visão computacional no escopo dos sistemas de distribuição de energia;
- estudar em implementar técnicas de desenlvolvimento de software;
- desenvolver uma aplicação computacional capaz de detectar em imagens a presença de postes com suas inclinações;
- desenvolver aplicação computacional capaz de detectar chaves do tipo fusível e extrair informação de seu estado (aberto ou fechado);

1.3 METODOLOGIA

Será realizada uma revisão bibliográfica sobre e análise do estado sobre o que já foi feito sobre a detecção de postes e chaves em imagens, além de uma pesquisa sobre Inteligência Artificial com foco nas aplicações de detecção de objetos em imagens através da consulta a levantamentos e artigos publicados nas revistas e conferências de maior impacto da área nos últimos anos. Na sequência serão escolhidas as arquiteturas de Redes Neurais Artificiais que se aplicam ao problema.

Será levantado e/ou confeccionado um banco de dados contendo imagens de postes e chaves. Na sequência os dados serão manuseados com o objetivo de atenderem padrões de rotulação utilizados por cada tipo de rede.

Serão escolhidas métricas para análise dos resultados obtidos na etapa de treinamento que permitam com que se avalie a performance das da arquitetura de rede neural artificial utilizada.

Caso seja necessário, serão aplicadas técnicas para melhorar o banco de dados, bem como sua ampliação. Além disso, serão verificadas e implementadas, caso necessário, melhorias nas redes escolhidas.

1.4 ORGANIZAÇÃO DO TEXTO

Este documento está estruturado na forma de 5 capítulos.

No primeiro capítulo encontram-se a introdução ao projeto e ao seu tema, bem como a definição do problema que será endereçado ao longo do trabalho. Também se encontram os objetivos gerais e específicos, além da metodologia a ser utilizada.

No segundo capítulo encontra-se uma revisão teórica que traça uma linha de desenvolvimento temporal sobre a detecção de postes e chaves em imagens, além de problemas correlatos com implementações que podem dar suporte ao trabalho.

No terceiro capítulo serão introduzidos e fundamentados os métodos de aprendizado profundo utilizados na tarefa de detecção de objetos em imagens, focando nas redes neurais do tipo convolucional.

O quarto capítulo traz os registros da etapa de desenvolvimento do projeto técnico além da avaliação dos resultados obtidos.

Por fim, no quinto e último capítulo, encontram-se as conclusões do projeto, além das discussões sobre os trabalhos futuros a serem desenvolvidos.

2 REVISÃO TEÓRICA

Nesta seção serão apresentados os resultados obtidos para pesquisas realizadas através do sitio *Google Scholar*, do portal de periódicos Capes, o programa de pesquisa acadêmica *Publish or Perish*, do sitio *Science Direct* e da biblioteca digital do Instituto de Engenheiros Eletricistas e Eletrônicos (em inglês, *Institute of Electrical and Electronics Engineers* - IEEE) com a temática central da detecção de postes e chaves em imagens.

2.1 HISTÓRICO

Embora seja possível encontrar exemplos na literatura, não há uma vasta gama de publicações tratando especificamente do problema da detecção de postes em imagens e tão pouco que trate especificamente da detecção de seus ângulos e componentes como, por exemplo, o caso das chaves fusível.

Um dos primeiros trabalhos sobre o tema que se utiliza de visão computacional foi realizado por D. I. Jones (2000), que ao tratar da inspeção de linhas elétricas aéreas utilizando vídeo acaba também por englobar a detecção de postes. Mais tarde, Jones, Whitworth e Duller (2003) se dedicam a aprofundar esta aplicação. Os autores apresentam um algoritmo para inspeção que de maneira rudimentar é capaz de localizar o posicionamento dos postes se utilizando da técnica de processamento digital de imagens conhecida como casamento de padrões. Está técnica, contudo, apresentava na época um alto custo computacional, baixa acurácia e não era adaptável aos diferentes modelos de postes.

Golightly e Dewi Jones (2003), Golightly e Dewi Jones (2005) e Dewi I. Jones et al. (2005) ao tratar do problema de automatização da inspeção de linhas elétricas aéreas propõem um método de detecção para postes baseado na detecção dos contornos, o que resultou em uma melhoria na tarefa. Contudo, resultados insatisfatórios eram obtidos em função da variação da iluminação e dos padrões no plano de fundo das imagens.

Sendo assim, W. Cheng e Song (2008) propõem uma abordagem para a detecção de postes em imagens que se baseia na técnica de graph-cut para segmentação de imagens. Nesta técnica a imagem é vista como um grafo ponderado, no qual os pixels da imagens representam os nós. Para reduzir o custo computacional, a princípio a imagem passa por uma filtragem na qual as linhas retas são detectadas, define-se uma região em que o poste pode vir a se encontrar e na sequência a construção do grafo ponderado. Na época, esta técnica apresentou uma melhora nos resultados que

até então haviam sido obtidos.

Podemos também extrapolar e observar outras aplicações que são relacionadas ao problema, como é o caso da detecção de postes em imagens aéreas (CETIN; BIKDASH; MCINERNEY, 2009), que se utiliza da detecção da sombra dos postes e de técnicas como casamento de padrões e extração de características.

Outro problema correlato é a deteção de torres de transmissão de energia em imagens (TILAWAT; AUEPHANWIRIYAKUL; THEERA-UMPON, 2010) e (LI et al., 2008). Ambos os trabalho citados fazem uso da Transformada de Hough, sendo que o primeiro se utiliza posteriormente de um filtro IIR e o segundo de um processo de clusterização dos dados.

Steiger, Lucas e Maret (2014), por sua vez, separam sua aplicação em fases de treino e teste. No treino, realiza-se a extração de características através do Line Segment Detector, a clusterização dos dados e a obtenção das distribuições espaciais. Na detecção, realiza-se a extração de características da imagem, a correspondência com o modelo e um processo de votação, tomando como comparação os valores obtidos com os do modelo de treino, e desta forma localizando o objeto de interesse.

Ainda tratando da detecção de torres de transmissão de energia elétrica em imagens, Sampedro et al. (2014) e Martinez et al. (2014) se utilizam de uma abordagem que faz uso do HOG para extração de características e da ANN do tipo perceptron multicamadas (do inglês, *Multilayer Perceptron - MLP*) para classificação.

Além disso, também observa na literatura o interesse em se realizar a detecção automática de postes em ambientes urbanos, como vemos no trabalho de Tombari et al. (2014) que se utilizam de um pré-processamento dos dados, extração de características, máquina de vetor de suporte (em inglês *support vector machine* - SVM) e clusterização para obter os resultados.

Voltando à detecção de postes, Sharma et al. (2015) propõem uma metodologia com shape-based template que é capaz de retornar resultado mesmo com o poste estando inclinado. Contudo, esta abordagem fica restrita a apenas um tipo de poste - o que se assimila ao template.

A detecção de postes também é vista na temática dos veículos autônomos, como em Blaga e Nedevschi (2018), que se utiliza da técnica de stereo-vision para realizar a tarefa.

Já se utilizando das técnicas de aprendizado profundo, vemos o trabalho de Nordeng et al. (2017) no qual os autores se utilizam da rede neural convolucional do tipo Faster-RCNN, além de téncnicas de data augmentation, para realizar a detecção de componente com corpo com pontas sem saídas em torres de transmissão de energia.

Através de imagens oriundas do sitio *Google Street View*, da utilização da rede do tipo one-stage detector RetinaNet, Zhang et al. (2018) conseguem realizar a identificação e localização de postes com cruzeta em imagens apresentando bons resultados.

Outra aplicação que também faz uso de uma rede do tipo one-stage detector é a vista no trabalho de Chen e Miao (2019). Os autores se utilizam da rede YOLO para realizar a detecção e contagem de postes em linhas de transmissão através de imagens obtidas com veículo aéreo não tripulado. Assim como a aplicação anterior, este tipo de rede se monstra vantajosa pela sua capacidade de realizar a tarefa de detecção de maneira veloz.

Como é possível observar, boa parte dos trabalhos publicados se utilizam de técnicas clássicas de processamento digital de imagens, sendo que em vários casos estas estão somadas a técnicas de aprendizado de máquinas para extração de características, clusterização, classificação e regressão. Contudo, desde o surgimento do aprendizado profundo (LECUN; BENGIO; G. HINTON, 2015), e das arquiteturas de redes neurais artificiais que se desdobraram desde então, a tarefa de detecção de objetos em imagens tem se voltado quase que integralmente para estas técnicas. Este fato, que pode ser observado nos exemplos anteriormente apresentados, pode ser averiguado também através de um extenso levantamento sobre o deselvolvimento da área de detecção de objetos em imagens que se deu nos últimos 20 anos realizado por Zou et al. (2019).

3 FUNDAMENTAÇÃO TEÓRICA

Neste cápitulo ocorre a fundamentação teórica do projeto. Serão apresentadas as técnologias que tiveram inspiração no cérebro com o objetivo de construir uma máquina que apresentasse inteligência e que hoje são utilizadas para a detecção de objetos em imagens, entre outros escopos.

3.1 REDES NEURAIS CONVOLUCIONAIS

Redes Neurais Artificiais (do inglês, *Artificial Neural Networks* - ANN) são estruturas computacionais inspiradas pelo funcionamento das redes neurais[1] dos seres vivos. Assim como o cérebro humano, a rede aprende e faz descobertas através de bons exemplos e de seus erros e correções, se tornando capaz de adquirir conhecimento através da experiência. Desta forma, o modelo computacional é capaz de executar tarefas de classificação diretamente a partir de imagens, textos ou sons.

Modelos de aprendizado profundo tem atingido uma precisão de estado da arte e por isso tem sido muito aplicados nas áreas de visão computacional, aprendizado de máquina e reconhecimento de padrões. Um dos tipos mais populares de ANN do tipo profundo são as Redes Neurais Convolucionais (do inglês, Convolutional Neural Networks - CNN), uma vez que são capazes de extrair sozinhas características dos dados de entrada, eliminando a necessidades de realizar esta tarefa manualmente. Para isto, elas se utilizam de camadas convolucionais 2D, o que as tornam adequadas para processar dados como os de imagens, por exemplo.

Nesta seção será realizada uma introdução à teoria sobre a estrutura e funcionamento das redes neurais artificiais através de uma abordagem que irá se focar posteriormente nas redes neurais do tipo convolucional.

3.1.1 Redes Neurais Artificiais

O cérebro humano é um sistema de processamento paralelo de informações altamente complexo e não linear. Ele possui a capacidade de organizar seus componentes estruturais, conhecidos como neurônios, de maneira a ser capaz de realizar em altíssima velocidade tarefas como reconhecimento de padrões, percepção e controle motor, por exemplo.

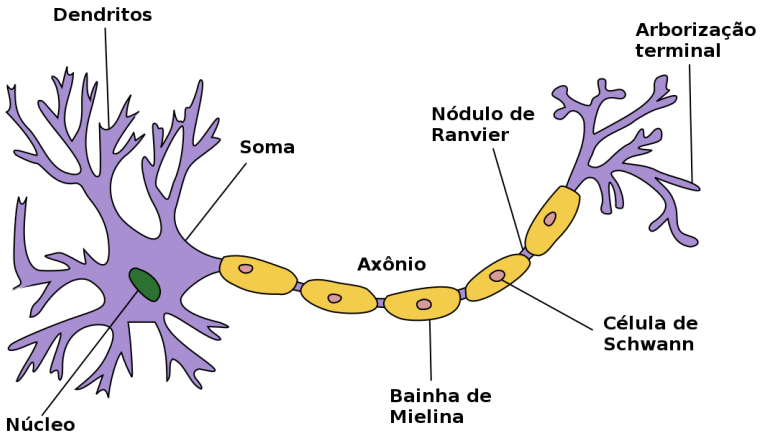
[1] As redes neurais artificiais são inspiradas no funcionamento das **redes neuronais** dos seres vivos, ou seja, dos neurônios. Contudo, na literatura da área em lingua portuguesa é mais comum encontrar o termo **rede neural** para este propósito, o qual pode ter um significado mais amplo mas na prática é utilizado no mesmo sentido.

Sendo assim, de acordo com Haykin (2008), rede neural artificial é uma 'máquina adaptativa', seja em forma de componentes eletrônicos ou de software, com uma modelagem inspirada pelo comportamento do cérebro, a qual tem como objetivo realizar determinadas tarefas ou funções. Assim como no cérebro, o conhecimento é adquirido a partir do meio através de um processo conhecido por aprendizagem e é armazenado através dos pesos entre as interconexões dos neurônios.

3.1.1.1 Perceptron

O neurônio é um tipo específico de célula que pode ser encontrada majoritariamente no córtex cerebral dos animais e que tem como função conduzir impulsos nervosos. Como pode ser visto na FIGURA 1, em sua composição três regiões se destacam: o corpo celular, onde se localiza o núcleo; os dendritos, que se conectam com os outros neurônios através das chamadas sinapses; e o axônio, uma estrutura alongada responsável por transmitir os impulsos para outros tipos de células. Quando um neurônio recebe um número suficiente de impulsos oriundos de outros neurônios, ele se ativa, transmitindo este sinal. Embora seu comportamento pareça simples, os neurônios se organizam aos bilhões, sendo cada um conectado a milhares de outros neurônios. Desta forma, o sistema como um todo é capaz de realizar cálculos complexos em altíssima velocidade.

FIGURA 1 – O NEURÔNIO

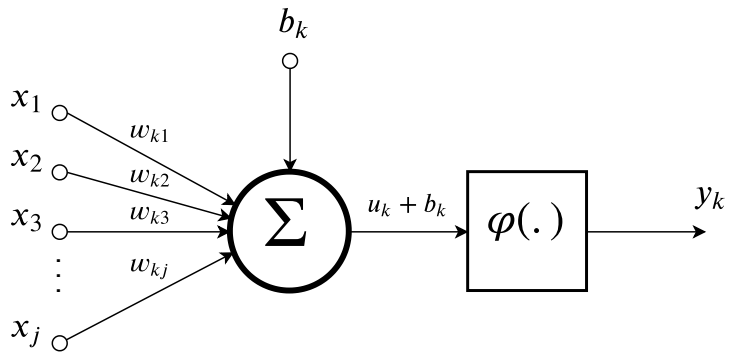


FONTE: Adaptado de Quasar Jarosz (2009).
LEGENDA: Estrutura típica de um neurônio.

Tomando como base o neurônio biológico surge o modelo de neurônio artificial conhecido como perceptron, o qual foi proposto por Rosenblatt (1958) baseado no modelo não linear de neurônio artificial desenvolvido por Mcculloch e Pitts (1990).

Em síntese, um perceptron é um modelo matemático com capacidade de representar em termos de algoritimo um comportamento que toma como base o de um neurônio biológico, sendo composto por três elementos básicos, os quais podem ser observados na FIGURA 2, abaixo.

FIGURA 2 – O PERCEPTRON



FONTE: O autor (2019)
LEGENDA: Diagrama de funcionamento do perceptron.

O primeiro elemento é um conjunto de sinapses, sendo que cada uma possui um peso próprio. Desta forma, temos um sinal de entrada x_j, aplicado a uma sinapse j, a qual está conectada a um neurônio k, sendo este sinal multiplicado pelo peso da sinapse w_{kj}.

O segundo elemento é o somador, denominado combinador linear, no qual os valores obtidos pela multiplicação entre os sinais de entrada em cada sinapse pelos respectivos pesos de cada uma destas conexões são combinados. Sendo assim, o somatório das entradas ponderadas u_k é dado por:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \tag{3.1}$$

Há ainda um termo b_k conhecido como *bias*, ou limiar, aplicado diretamente ao combinador linear. Sua função é provocar um deslocamento no terceiro elemento básico de um neurônio artificial, a função de ativação $\varphi(.)$. Este terceiro elemento é o que definirá o valor de saída do neurônio tomando como base os valores resultantes do combinador linear. No caso do perceptron, esta função é conhecida por threshold

(3.2), onde **w** é o vetor dos pesos, **x** é o vetor com os valores da entrada das sinapses e **b** é o vetor com os valores dos limiares.

$$f_thresh(x) = \begin{cases} 1 & \text{se} \quad \mathbf{w}^T.\mathbf{x} + b > 0, \\ 0 & \text{caso contrario.} \end{cases} \tag{3.2}$$

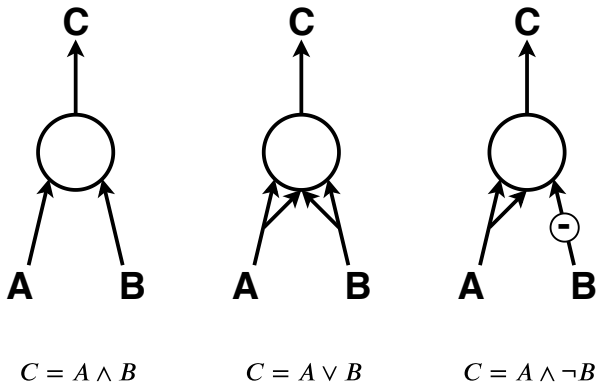
Sendo então $u_k + b_k$ a entrada de uma função de ativação, teremos que a saída do neurônio k, dada por y_k, é:

$$y_k = \varphi(u_k + b_k) \tag{3.3}$$

Desta forma, ao fazer $\varphi(.)$ como (3.2), um perceptron se torna um classificador linear binário, uma vez que é capaz de traçar uma fronteira entre duas classes de dados baseado em suas características.

Além disso, dependendo de sua configuração, o perceptron é capaz de realizar operações como a de portas lógicas, como pode ser visto na FIGURA 3. Suponha que neste caso as entradas A e B tenham valor 1 ou 0. Suponha também que cada seta a partir das entradas em direção ao círculo, que representa o somador, tenham valor 1. A função de ativação do neurônio será tal que ele irá se ativar (apresentar valor 1 na saída) quando o somatório dos valores de sua entrada for maior ou igual a 2. É possível observar como é possível computar as funções apresentadas para C através das combinações realizadas para estabelecer os pesos das sinapses.

FIGURA 3 – FUNÇÕES LÓGICAS



FONTE: Adaptado de Géron (2017).
LEGENDA: Funções lógicas computadas por neurônios artificiais.

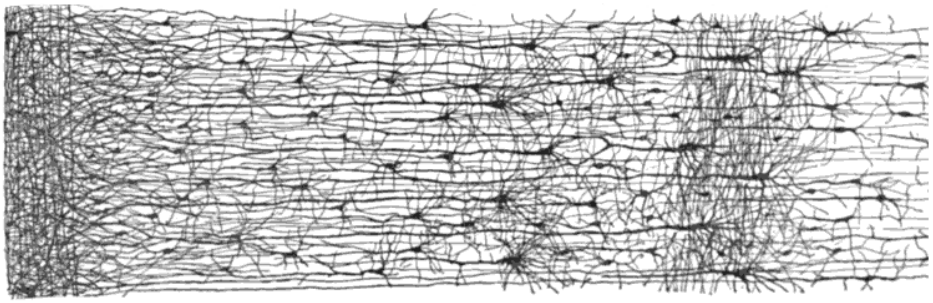
É comum na literatura ver todo neurônio artificial ser tratado pelo termo perceptron. Contudo, um neurônio artificial pode possuir outros tipos de função de ativação,

as quais serão exploradas com mais profundidade adiante, na seção 3.1.3.4. Outros termos comuns para o neurônio artificial na literatura são nó e unidade de processamento.

3.1.1.2 Perceptron multicamadas

Ainda há muita pesquisa em desenvolvimendo na busca de mapear as diferentes regiões do cérebro. Contudo, até o momento, tem se percebido que frequentemente em redes neurais biológicas os neurônios estão organizados em camadas consecutivas, como pode ser visto na FIGURA 4, abaixo.

FIGURA 4 – CORTEX



FONTE: Santiago Ramon e Cajal (1899)
LEGENDA: Cortéx cerebral de uma criança de 1 ano e meio obtido com o método de Golgi.

Assim como no neurônio biológico, o neurônio artificial também pode ser organizado desta forma. O perceptron multicamadas (do inglês, *Multilayer Perceptron* - MLP) é uma estrutura que pode ser dividida em três blocos, como pode ser visto na FIGURA 5.

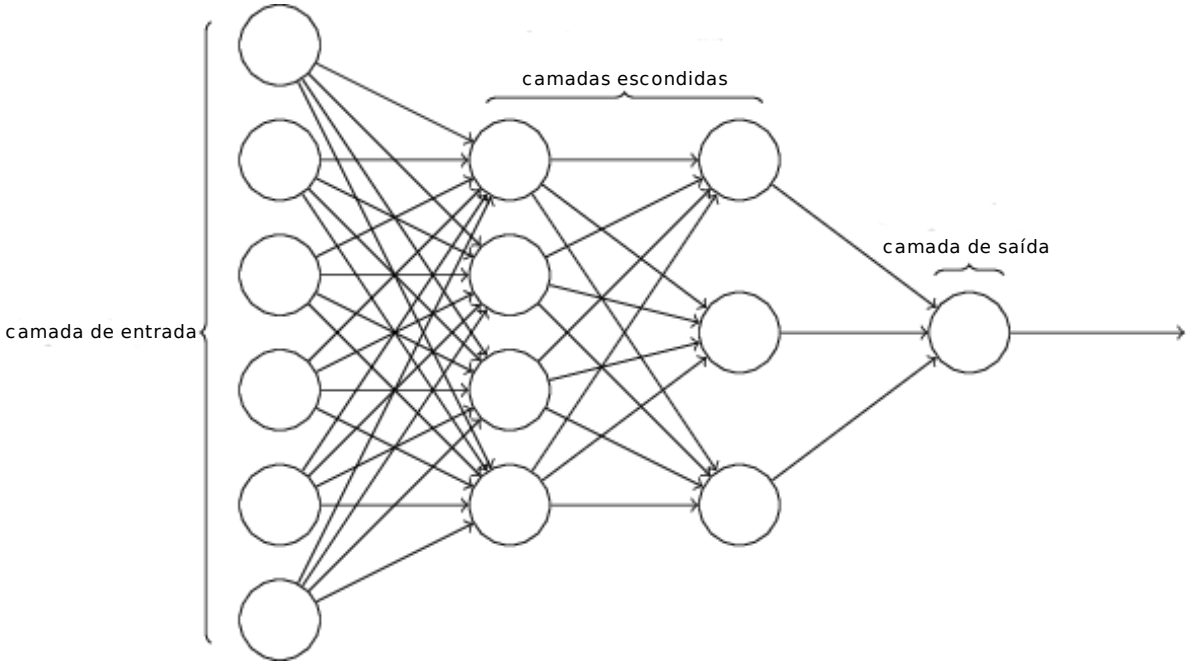
O primeiro bloco do MLP é a camada de entrada, a qual contém os sinais aplicados à rede. Esta representação, na qual estes valores aparecem como círculos e não da forma representada na seção 3.1.1.1 é a mais comumente encontrada na literatura quando mais de uma camada é representada. As entradas de limiar também são suprimidas em nome da simplificação da representação. É importante destacar que todos os neurônios artificiais possuem um valor de limiar, exceto os da última camada - a de saída.

O segundo bloco é o das camadas escondidas. De acordo com Haykin (2008), estas são as camadas que permitem que a rede aprenda tarefas complexas por extrairem progressivamente características significativas dos padrões do vetor das entradas. É importante ressaltar que cada perceptron possui apenas um valor de saída.

A representação da FIGURA 5 é utilizada para mostrar que este valor único de saída é aplicado à entrada de todos os neurônios da camada seguinte.

O terceiro bloco é o da camada de saída. É neste bloco que se apresenta o valor do resultado obtido para um dado vetor de valores de entrada.

FIGURA 5 – MULTILAYER PERCEPTRON



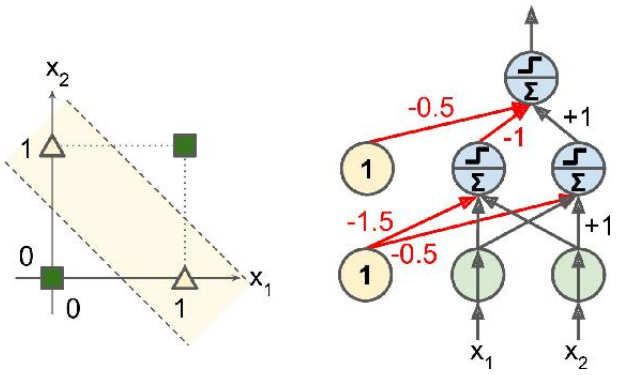
FONTE: O autor.
LEGENDA: Diagrama da estrutura do MLP.

Como visto anteriormente, com um único perceptron é possível realizar funções como AND e OR, e separar linearmente dois tipos de dados com características distintas. Contudo, em um problema um pouco mais complexo, como o caso do XOR, apenas um perceptron não é capaz de resolver o problema. Com a adição de apenas uma só camada escondida, como pode ser visto na FIGURA 6, já se torna possível encontrar uma função capaz de resolver o problema.

Não há limite quanto ao número de neurônios por camada nem quanto ao número de camadas escondidas que uma rede do tipo MLP pode ter. Estes valores podem variar de acordo com o problema ou aplicação da rede.

Uma vez que diferentes funções de ativação podem ser aplicadas ao neurônio artificial – funções que inclusive adicionam não-linearidade à ele – o MLP dá origem ao

FIGURA 6 – XOR



FONTE: Géron (2017)
LEGENDA: Função XOR computada com MLP

que conhecemos por rede neural artificial. E se esta possuir duas ou mais camadas escondidas, de acordo com Géron (2017), ela recebe o nome de rede neural profunda (do inglês, Deep Neural Network - DNN).

Redes do tipo DNN tem sido muito utilizadas nos últimos anos para resolver diversos problemas de várias complexidades e com muito sucesso. Para aprenderem, estas redes passam por um processo conhecido como treinamento e que faz uso de um algoritmo conhecido por retropropagação, o qual será aprofundado no item 3.1.2.3, na seção a seguir.

3.1.2 Aprendizado em Redes Neurais Artificiais

O que determina os valores de saída de uma ANN para uma dada entrada é a relação entre os pesos atribuídos em cada uma das conexões entre os neurônios, o valor dos limiares e a função de ativação, que é utilizada para adicionar a capacidade de não-linearidade ao modelo da rede. Sendo assim, treinar uma ANN signfica aprender quais são os valores mais ajustados para estes parâmetros (de w e b), através de um processo interativo.

No processo de aprendizado supervisionado (item 3.1.2.4), dados rotulados são apresentados à rede. Na sequência, a rede irá propagar valores para os pesos e limiares de cada unidade, camada a camada. Estes valores iniciais costumam ser aleatórios. Sendo assim, a rede irá obter um valor de saída e compará-lo com os dados rotulados que foram submetidos à rede. Para isto, será utilizada uma função de custo (item 3.1.2.2) para estimar e medir o erro entre os valores de entrada e os valores inferidos.

Na sequência, o valor do erro calculado será propagado pela rede no sentido inverso, ou seja, da última camada em direção à primeira, em um processo conhecido

por retropropagação (item 3.1.2.3). Como cada unidade irá contribuir com uma fração do erro total, os pesos de suas conexões sofrerão pequenos ajustes utilizando o método do gradiente (item 3.1.2.1), que tem como objetivo fazer com que os valores sejam ajustados de maneira a levar a função de custo para seu mínimo global.

Cada iteração deste processo, que irá se repetir por várias vezes proporcionando a rede aprender melhores valores para os pesos entre as conexões, recebe o nome de época (do inglês *epoch*). Observando-se os valores dos erros para cada época do treinamento é possível averiguar se a rede está convergindo (aprendendo) ou divergindo, dessa forma possibilitando saber em quantos passos o treinamento pode ser interrompido.

Por fim, dados nunca antes vistos pela rede a ela são apresentados. A partir daí utilizam-se técnicas para gerar métricas (3.1.2.7) para avaliar a qualidade do modelo.

3.1.2.1 Método do gradiente

Dadas as características da arquitetura de uma ANN, tem-se por objetivo obter um algoritmo capaz encontrar valores para os pesos entre as conexões dos neurônios, bem como dos limiares, para um dado conjunto de dados de treino x aplicados à rede. Além disso, estes valores obtidos pelo algoritmo devem restultar em uma $y(x)$ para a saída da rede que seja aproximado àquele que se espera como resultado.

Para que este objetivo seja atingido, defini-se uma função conhecida por função de custo[2], dada por $C(w, b)$, onde w representa o conjunto de todos os pesos de uma rede e b conjunto de todos os valores de limiar de uma rede.

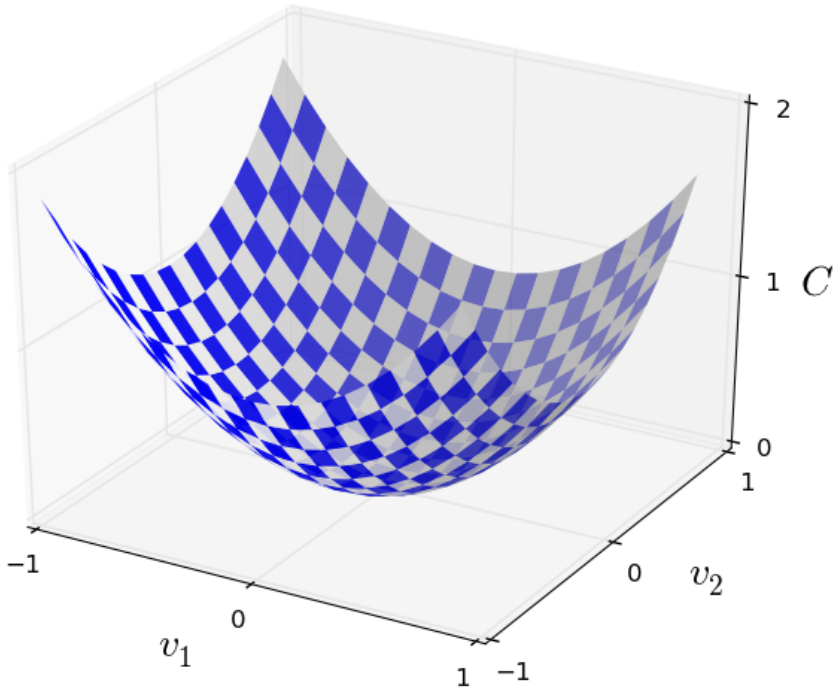
Existem várias funções de custo que podem ser utilizadas, que serão apresentadas no item 3.1.2.2, mas uma característica em comum entre elas é que $C(w, b)$ é positiva. Desta forma, a saída $y(x)$ estará mais próxima de x quanto mais próximo estiver o seu valor de 0.

Sendo assim, o objetivo do algoritmo de treino da ANN é minimizar a função $C(w, b)$ através dos valores de w e b, de modo que o custo seja o menor possível. O algoritmo que executa esta tarefa recebe o nome de método do gradiente.

De acordo com Nielsen (2015), uma maneira mais fácil de visualizar o problema é fazendo a função custo como $C(v)$, onde v é uma função com valor real de várias variáveis, $v = v_1, v_2, ...$ Abaixo, na FIGURA 7, utiliza-se v sendo uma função de duas variáveis, v_1 e v_2, como exemplo.

[2] Também conhecida como função de perda ou função objetiva.

FIGURA 7 – VALE



FONTE: Nielsen (2015)
LEGENDA: vale

Deseja-se encontrar o mínimo global da função custo, o que neste exemplo é fácil de se fazer visualmente. Contudo, o problema é muito mais complexo na realidade, uma vez que envolverá muitas variáveis além de diferentes comportamentos que resultarão em inúmeros mínimos locais. Porém, este exemplo é útil para se compreender a base teórica do algoritmo do método do gradiente.

O algoritmo inicia em um ponto aleatório da superfície, que neste ponto pode ser pensada como um vale. É útil imaginar que neste ponto será colocada uma bola. A tendência é esta bola rolar em direção ao mínimo da função.

Este movimento pode ser pensado da seguinte forma: sejam Δv_1 e Δv_{v2} um pequeno movimento na direção v_1 e v_2, respectivamente, o movimento é dado por:

$$\Delta C \approx \frac{\partial C}{\partial v_1}\Delta v_1 + \frac{\partial C}{\partial v_2}\Delta v_2 \tag{3.4}$$

Uma vez que espera-se que ΔC mova-se em direção ao mínimo, buscam-se valores de Δv_1 e Δv_2 que façam ΔC ser negativo.

Para encontrar este valor, primeiro será definido Δv como o vetor das mudanças em v, onde aplica-se a operação de transposição para transformar os vetores em colunas.

$$\Delta v \equiv (\Delta v_1, \Delta v_2)^T \tag{3.5}$$

Além disso, o gradiente de C é definido como o vetor das derivadas paraciais de v, que também será transposto. Sendo assim, o gradiente de C equivale a:

$$\nabla C \equiv \left(\frac{\partial C}{\partial v_1}, \frac{\partial C}{\partial v_2} \right)^T \tag{3.6}$$

Reescrevendo ΔC em função das equações 3.6 e 3.5, temos que:

$$\Delta C \approx \nabla C \cdot \Delta v \tag{3.7}$$

A equação 3.7 nos permite ver como é possível escolher um valor de Δv que faça com que ΔC seja negativo. Por exemplo, suponha que:

$$\Delta v = -\eta \nabla C, \tag{3.8}$$

onde η é um valor pequeno e positivo. Sendo assim, a equação 3.7 ficará como $\Delta C \approx -\eta \nabla C \cdot \nabla C = -\eta \left\| \nabla C \right\|^2$. Sendo $\left\| \nabla C \right\|^2 \geq 0$, isto garante que $\Delta C \leq 0$, fazendo com que a função custo sempre decresça, dentro dos limites da aproximação da equação 3.7.

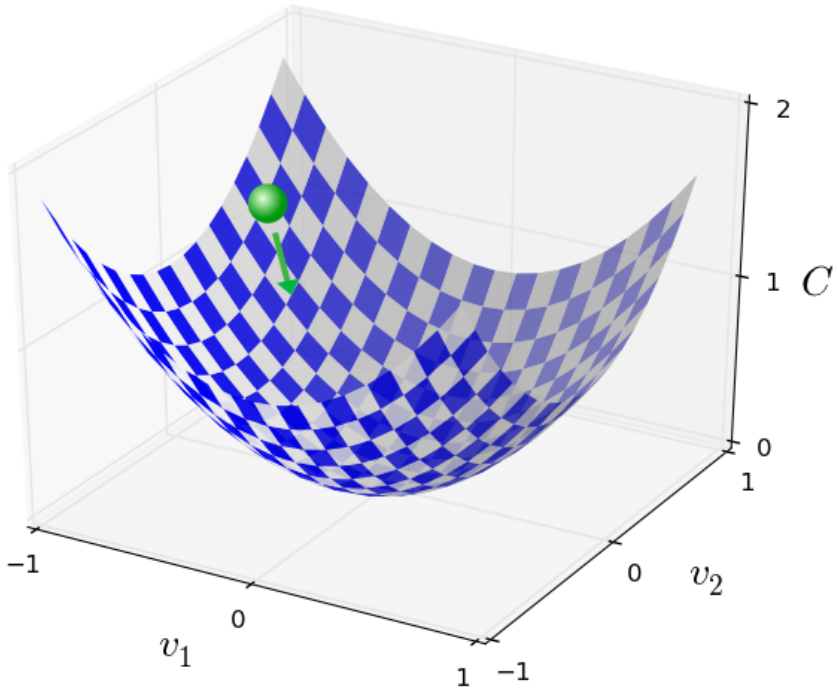
Sendo então a equação 3.8 a que define o ”movimento” da bola na superfície apresentada na FIGURA 7, o movimento de v para v', será:

$$v \rightarrow v' = v - \eta \nabla C \tag{3.9}$$

A cada iteração do processo de treinamento da ANN, espera-se então que mais um pequeno passo seja dado em direção ao mínimo global da função custo ou próximo a isto. A FIGURA 8 exemplifica este movimento em direção ao ponto de mínimo da função.

Embora este algoritmo funcione muito bem na prática, é possível que o mínimo global não seja localizado por ele. Além disso, dada a complexidade que surge de uma ANN, que pode possuir bilhões de parâmetros, além de diversas funções de ativação, é

FIGURA 8 – VALE BALL



FONTE: Nielsen (2015)
LEGENDA: vale com bola

possível que nem sempre o passo dado seja em direção ao mínimo. Segundo Nielsen (2015), pode-se pensar nesta bola do exemplo como possuindo um momento, que a fará rolar através da encosta da superfície e até, momentaneamente, subir esta encosta, mas sempre tendendo em direção ao mínimo da função.

Outro ponto importante para o funcionamento deste algoritmo é a escolha do parâmetro η, também conhecido como taxa de aprendizado. Este parâmetro pode ser visto como o tamanho do passo que a função dará a cada iteração do algoritmo de treinamento da ANN.

O valor da taxa de aprendizado deve ser pequeno o suficiente para que a equação 3.7 seja uma boa aproximação e não acabe com $\Delta C > 0$, o que faria o algoritmo de aprendizado divergir uma vez que a função de custo não iria se aproximando em direção ao seu mínimo, e nem excessivamente pequeno, o que faria o movimento do algoritmo em direção ao mínimo global muito pequeno, fazendo com que o método do gradiente funcione muito lentamente, exigindo muitas iterações para a rede convergir. Outro problema com uma taxa de aprendizado muito pequena é que pode fazer com

que o algoritmo fique preso a um mínimo local.

Dadas estas características, uma ANN irá aprender através da aplicação do método do gradiente a uma dada função custo através de um processo iterativo, pelo qual serão encontrados valores para w_k e b_l que minimizam esta função. Escrevendo então o método do gradiente em função dos componentes da ANN temos que:

$$w_k \rightarrow w'_k = w_k - \eta \frac{\partial C}{\partial w_k} \tag{3.10}$$

$$b_l \rightarrow b'_l = b_l - \eta \frac{\partial C}{\partial b_l} \tag{3.11}$$

3.1.2.2 Funções de custo

Existem vários tipos de função de custo, sendo que sua escolha dependerá do tipo de problema com o qual está se tratando.

Os principais tipos de problema são e suas funções de custo mais populares são: classificação binária (*SVM hinge, Aquared hinge*), verificação de identidade (*Constrastive*), classificação multi-classe (*Softmax, Expectation*) e regressão (SSIM, Euclideana) (KHAN et al., 2018).

Aqui focaremos em apresentar a função Softmax, que é uma das mais populares de função de custo para o tipo de problema que aparece neste projeto.

Softmax – A função Softmax pode ser vista como um classificador. Não exclusivamente, mas ela geralmente aparece ao final da ANN. Ela recebe um valor y_i de saída da rede neural e mapeia a probabilidade daquela saída pertencer a cada classe do problema.

$$S(y_i) = \frac{e^{y_i}}{\sum_j e^{y_j}} \tag{3.12}$$

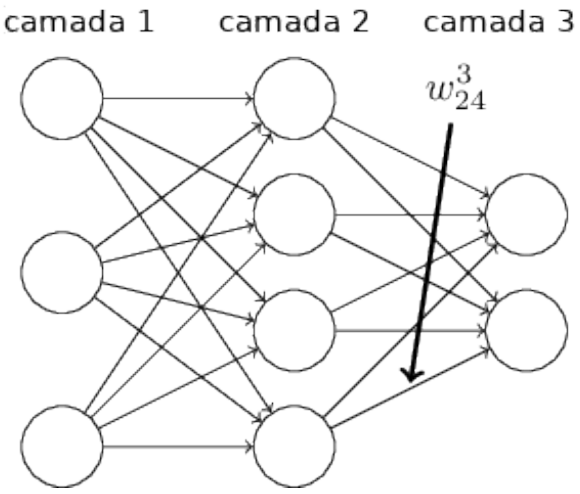
3.1.2.3 Retropropagação

Originalmente introduzido por Rumelhart, G. E. Hinton e Williams (1986) na temática das ANNs, o algoritmo de retropropagação é o meio pelo qual se realiza a computação do gradiente da função custo. Este algoritmo fornece uma compreensão detalhada de como a mudança nos pesos e limiares alteram o comportamento como um todo da rede neural e permite computar as derivadas parciais da função custo C de qualquer peso $\partial C / \partial w^l_{jk}$ e de qualquer limiar $\partial C / \partial b^l_j$ pertencente à rede.

Será utilizada a notação sugerida por Nielsen (2015), onde w^l_{jk} representa o peso entre a conexão do k-ésimo neurônio da $(l-1)$-ésima camada com j-ésimo

neurônio na j-ésima camada. Além disso, b^l_j representa o limiar para o j-ésimo neurônio da l-ésima camada. Por fim, a^l_j representa a ativação do j-ésimo neurônio da l-ésima camada. A FIGURA 9 exemplifica esta notação.

FIGURA 9 – RETROPROPAGAÇÃO



FONTE: Adaptado de Nielsen (2015).
LEGENDA: $w^l_{jk} = w^3_{24}$, onde $j = 2$ é o segundo neurônio da terceira camada $l = 3$, que recebe a saída do quarto neurônio $k = 4$ da camada anterior $l - 1 = 3 - 1 = 2$.

Então, aplicando esta notação à equação 3.3, temos que:

$$a^l_j = \sigma\left(\sum_k w^l_{jk} a^{l-1}_k + b^l_j\right) \tag{3.13}$$

Por uma questão de simplificação a expressão 3.13 será reescrita em forma matricial, a qual pode ser verificada na equação 3.14, a seguir. Nesta equação, w^l representa a matriz dos pesos de entrada dos neurônios de uma camada l, b^l representa o vetor dos valores dos limiares de uma camada l e a^l o vetor dos valores da ativação dos neurônios de uma camada l. Desta forma, temos que:

$$a^l = \sigma(w^l a^{l-1} + b^l) \tag{3.14}$$

Esta expressão nos permite observar como as ativações de uma camada de neurônios se relaciona com as ativações da camada anterior. O termo aplicado à função de ativação recebe o nome de entrada ponderada (em inglês, *weighted input*), sendo

denotado por z^l, ou seja, $z^l \equiv w^l a^{l-1} + b^l$. Reescrevendo a equação 3.14 em termos de z^l, resulta-se em:

$$a^l = \sigma(z^l) \tag{3.15}$$

A partir de agora serão feitas duas suposições. A primeira é que a função custo pode ser escrita como a média das funções custo de cada exemplo de treinamento aplicado à rede, uma vez que o algoritmo de retropropagação computa as derivadas parciais da função custo em relação à w e b. Logo:

$$C = \frac{1}{n}\sum_x C_x, \tag{3.16}$$

onde x é um exemplo de treinamento individual, n é o número total de exemplos do treinamento e C_x é a função custo para um exemplo individual x.

A segunda é que a função custo pode ser escrita em função dos valores de saída da ANN, como:

$$C_x = C_x(a^L(x)), \tag{3.17}$$

onde $a^L(x)$ é o vetor com os valores de saída dos neurônios para a entrada x, e L denota o número total de camadas da rede neural. De agora em diante o subescrito x será suprimido por uma questão de simplificação.

Além disso, será introduzido δ^l_j, que é conhecido como o erro no j-ésimo neurônio na l-ésima camada, o qual poderá ser computado pelo algoritmo de retropopagação e será relacionado com $\partial C/\partial w^l_{jk}$ e $\partial C/\partial b^l_j$, que são necessários para a aplicação do método do gradiente.

Suponha que há uma pequena variação, dada por Δz^l_j no j-ésimo neurônio da l-ésima camada. Isto fará com que a saída do neurônio seja $\sigma(z^l_j + \Delta z^l_j)$. Esta variação se propagará por todas as camadas da rede, fazendo o custo total mudar por uma taxa $\frac{\partial C}{\partial z^l_j}\Delta z^l_j$. Sendo assim, através da escolha de um valor para Δz^l_j, busca-se melhorar o custo da rede. Por exemplo, se $\frac{\partial C}{\partial z^l_j}$ é um valor alto, é possível reduzir o custo através da escolha de um Δz^l_j com sinal oposto a este valor. Por outro lado, se Δz^l_j é próximo de zero, provoca-se pouca melhoria no custo através da perturbação causada em z^l_j , ou seja, este neurônio já está perto de seu ótimo. Em resumo, $\frac{\partial C}{\partial z^l_j}$ pode ser visto como uma medida do erro no neurônio j da camada l e é definido como:

$$\delta^l_j \equiv \frac{\partial C}{\partial z^l_j} \tag{3.18}$$

A partir daí será possível definir as quatro equações fundamentais da retropropagação.

A primeira é o erro na camada de saída:

$$\delta_j^l = \frac{\partial C}{\partial a_j^L}\sigma'(z_j^l), \tag{3.19}$$

que, da maneira matricial, fica como:

$$\delta^L = \nabla_a C \odot \sigma'(z^L), \tag{3.20}$$

onde $\odot$ é a multiplicação elemento por elemento da matriz, operação também conhecida por produto de Hadamard. O termo $\frac{\partial C}{\partial a_j^L}$ é uma medida do quão rápido o custo está mudando em função da j-ésima saída da ativação e $\sigma'(z_j^l)$ mede o quão rápido a função de ativação está mudando em z_j^L.

A segunda equação é o erro δ^l em termos do erro da camada seguinte δ^{l+1}.

$$\delta^L = ((w^{l+1})^T\delta^{l+1}) \odot \sigma'(z^l) \tag{3.21}$$

Esta é a função que permite retropropagar o valor do erro através da rede. Combinando ainda as equações 3.21 com 3.20 é possível computar o erro em qualquer camada da rede. Inicia-se calculando δ^L, que permite computar δ^{L-1}, δ^{L-2}, e assim por diante.

A terceira equação permite calcular a taxa de mudança no custo em função de qualquer valor de limiar da rede.

$$\frac{\partial C}{\partial b_j^l} = \delta_j^l \tag{3.22}$$

E, por fim, a quarta equação, que permite calcular a taxa de mudança no custo em função de qualquer peso entre os neurônios.

$$\frac{\partial C}{\partial w_{jk}^l} = a_k^{l-1}\delta_j^l \tag{3.23}$$

A demonstração matemática destas equações pode ser encontradas em Nielsen (2015).

Agora que se sabe como calcular as derivadas parciais e o erro em cada camada da ANN que permitem com que a rede aprenda, serão apresentadas na próxima seção (3.1.2.4) como é aplicado o procedimento de aprendizado.

3.1.2.4 Aprendizado supervisionado

No aprendizado supervisionado um conjunto de dados previamente rotulados é introduzido à rede. Estes rótulos apresentam à rede qual o valor de saída é esperado para os dados que a ela são introduzidos de acordo com suas características. Por exemplo, imagine um banco de dados contendo mensagens de duas classes, sendo a primeira a classe spam (quando o e-mail é não-solicitado e enviado pelo remetente para inúmeras pessoas) e segunda do tipo ham (quando o e-mail é do tipo solicitado), as características (*features*, em inglês) destes dados podem ser: comprimento da mensagem, frequencia de certas palavras, número de destinatários e assim por diante. Cada característica terá um atributo, que é um valor numérico associado à característica, por exemplo: comprimento da mensagem = 300 caracteres.

No campo do aprendizado de máquina clássico a anotação das características costuma ser decidida pelo desenvolvedor. Em aprendizado profundo, contudo, estas característas são decididas e extraídas automaticamente pela rede em seu processo de aprendizado. Como isto ocorre será abordado mais adiante. Porém, ainda é necessário selecionar a região de interesse em que se encontra a informação e sua classificação com o objetivo de afunilar a quantidade de dados que a rede precisa processar em seu aprendizado, tornando este mais direcionado. A estas bordas que envolvem o objeto de interesse, dá-se o nome de caixa de seleção (ou *bounding box*, do original em inglês).

A seguir, na FIGURA 10, temos um exemplo de uma imagem na qual é anotada a região em que se encontram as classes que nela estão presentes. Embora existam vários padrões de como registrar em arquivo estas anotações não é incomum que um desenvolvedor defina um padrão próprio que se encaixa melhor ao seu sistema.

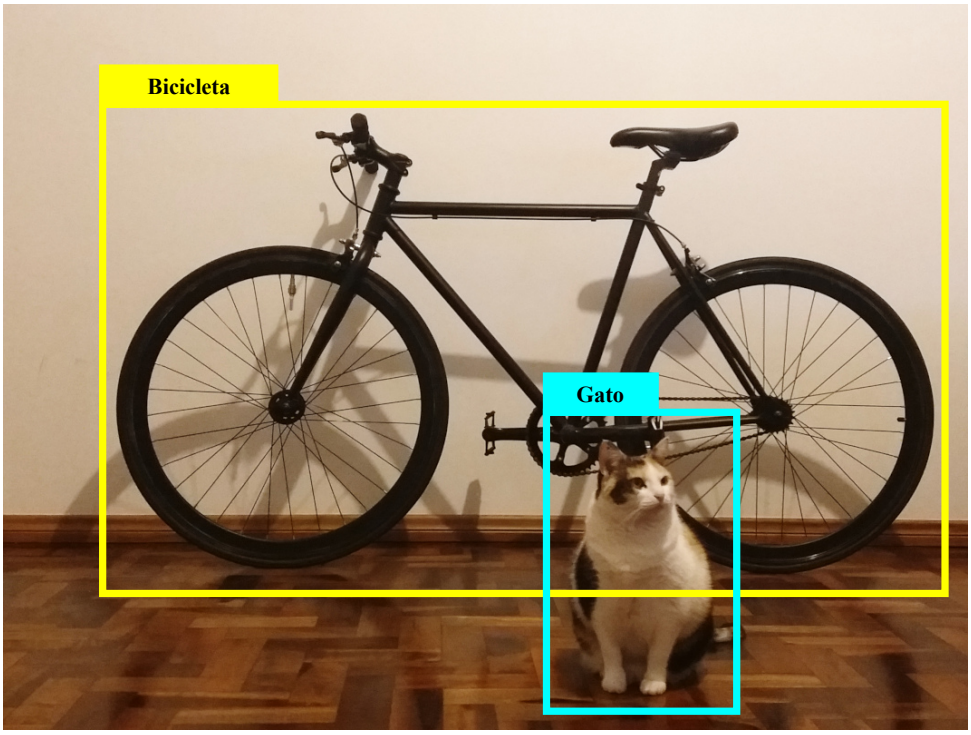
Existe também uma série de boas práticas que devem ser adotadas com relação à organização dos dados.

São três os conjuntos de dados que são utilizados no processo de aprendizado e avaliação de uma ANN: treino, validação e teste.

O maior conjunto de dados é o de treino, o qual é utilizado pela rede para aprender os pesos dos neurônios. O segundo, de validação, é utilizado para acompanhar o processo de treinamento. E o último, de teste, é um conjunto de dados nunca antes visto pela rede que é utilizado para o processo de avaliação do modelo.

É comum encontrar as seguintes divisões para os dados: 60% treino, 20% validação e 20% teste. Ou, quando o banco de dados é pequeno, 80% treino e 20% teste. Embora seja comum encontrar estas proporções na literatura da área de aprendizado de máquinas, outros valores podem ser utilizados.

FIGURA 10 – CAIXAS DE SELEÇÃO



FONTE: O autor.
LEGENDA: Caixas de seleção: em azul, em torno do objeto da classe gato e em amarelo, classe bicicleta.

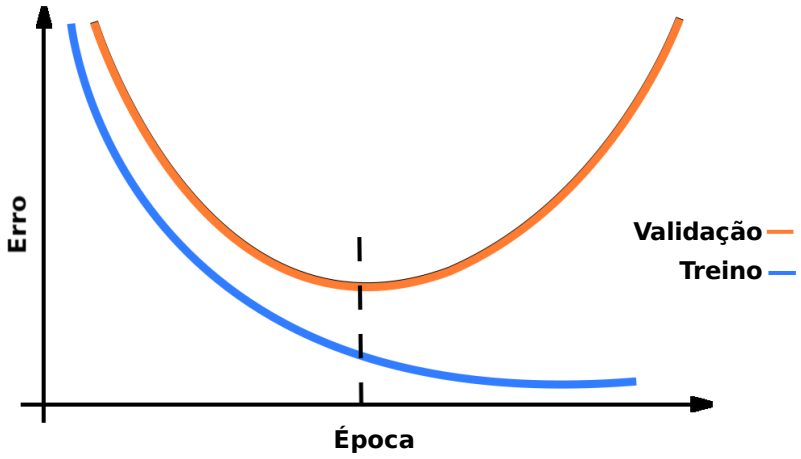
A FIGURA 11 mostra como ocorre o processo de treinamento utilizando dados de treino e validação. É possível observar como a rede vai obtendo uma boa generalização dos dados, ou seja, o erro (ou perda) para o conjunto de validação está próximo ao do treino, até que em certo ponto a rede passa a realizar o chamado *overfitting*, que é quando ao invés de aprender a generalizar a rede começa a decorar os dados do conjunto de treino. Onde os erros dos conjuntos de treino e de validação estivere mais próximos é o momento de parar o treinamento. Na sequência, neste ponto, aplica-se o conjunto de teste para avaliar a rede.

De acordo com Ozdemir (2017), uma das melhores formas de estimar a performance de um modelo é através da técnica de validação cruzada conhecida como k-fold.

Nesta técnica o conjunto de dados é dividido em k partes iguais, sendo este valor geralmente superior a 5. Cada uma dessas partes é um fold (que pode ser traduzido para o português como dobra).

Uma dessas dobras será retirada do conjunto para ser utilizada como conjunto de teste e as dobras restantes são utilizadas para o treino. Ao final do treino são

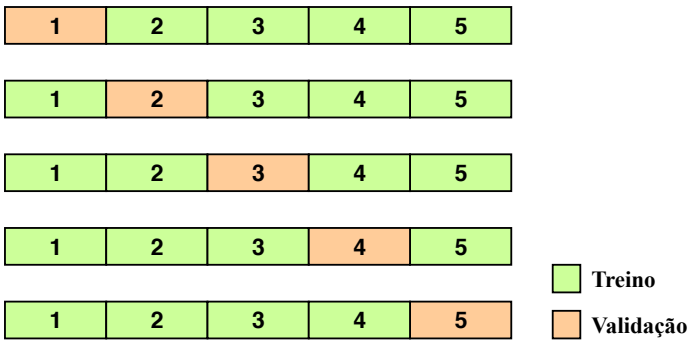
FIGURA 11 – CURVAS DE TREINO E VALIDAÇÃO



FONTE: Adaptado de Paweł Grabiński(2018)
LEGENDA: Em azul: curva de treino. Em laranja: curva de validação.
Tracejado: ponto de parada ideal do treinamento.

levantadas as métricas de avaliação do aprendizado (item 3.1.2.7). Na sequência esta dobra volta para o conjunto de dados e o próximo é retirado para ser agora o de teste. Assim, as estapas de treino e teste são realizadas até que todas as dobras tenham sido utilizadas para teste. Por fim, o resultado final do modelo é a média dos resultados obitidos para cada treinamento.

FIGURA 12 – VALIDAÇÃO CRUZADA K-FOLD



FONTE: O autor
LEGENDA: Validação cruzada com 5 dobras.

Esta técnica é muito útil pois permite averiguar se o conjunto de dados tem uma boa distribuição. Por exemplo, suponha que em um conjunto de dados de 100

imagens, sendo as 80 primeiras de gatos e as últimas 20 de cães, é separado em 80% treino e 20% teste. Porém, os dados separados para teste são justamente as últimas 20 imagens. A rede irá aprender as características de um gato, mas o teste só possui imagens de cães, uma classe antes nunca vista pela rede. O resultado não irá representar fielmente a capacidade da rede em aprender estas classes.

Com a técnica de validação cruzada, além de se ter uma maneira mais fiel de avaliar a rede, também é possível fazer observações com relação à distribuição das classes através do banco de dados.

Embora a validação cruzada seja um dos métodos mais indicados de avaliar a performace de um modelo de aprendizado de máquina, ela apresenta um problema que pode ser impeditivo no caso de bancos de dados muito grandes e redes muito complexas, que é o custo computacional, uma vez que o processo de treino e teste deve ser realizado várias vezes.

3.1.2.5 Aumento de Dados

Uma técnica bastante utilizada para melhoria da performance de uma ANN é a de aumento de dados (*data augmentation*, do termo original, em inglês).

O aumento de dados consiste em aplicar uma série de alterações em imagens, sejam elas aplicadas de maneira isolada ou em conjunto, como cortes, rotações, inversões horizontais e verticais, aproximações, distorções, ruídos, e assim por diante. Gerando um novo conjunto derivado das imagens originais. Este conjunto então passa a integrar o banco de dados original. A FIGURA 13 exemplifica este processo. À esquerda vemos a imagem original e o restante são imagens que dela foram derivadas.

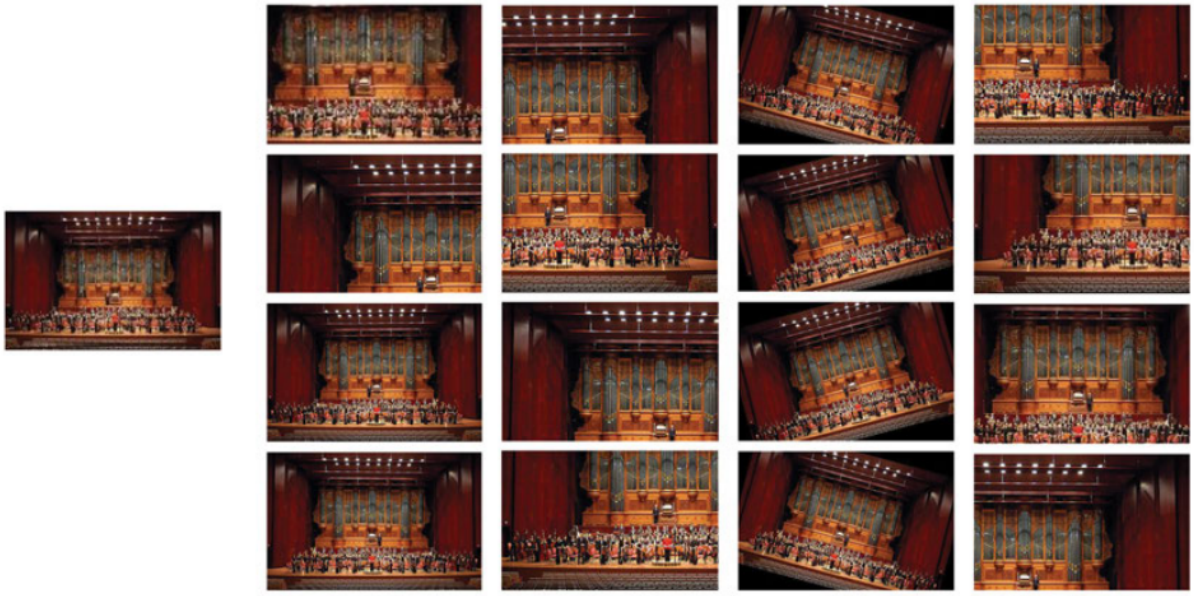
Esta técnica permite extrapolar o conjunto inicial de dados, criando uma variabilidade que ajuda a rede a gerar um modelo capaz de generalizar melhor o problema.

3.1.2.6 Transferência de Conhecimento

Os seres humanos têm uma capacidade inerente de transferir conhecimento entre tarefas. O que adquirimos como conhecimento enquanto aprendemos sobre algo, nós utilizamos da mesma maneira para resolver problemas que estão relacionados. Quanto mais relacionadas são as tarefas, mais fácil é para transferirmos ou utilizarmos de maneira cruzada nosso conhecimento prévio (SARKAR; BALI; GHOSH, 2018).

De maneira análoga, podemos utilizar o conhecimento adquirido por uma ANN em um domínio específico para outro que apresente alguma relação.

FIGURA 13 – AUMENTO DE DADOS



FONTE: Adaptado de Khan et al. (2018).
LEGENDA: À esquerda: imagem original. Demais imagens: resultado do técnica de aumento de dados.

O processo de treinamento de uma ANN, principalmente no caso de dados com alta dimensionalidade, como imagens, pode ser muito dispendioso computacionalmente. Por isso, é muito comum utilizar como ponto de partida para o treinamento os valores de pesos e limiares encontrados por outra rede, que já tenha passado por um longo processo de treino. Por exemplo, uma ANN é construída para reconhecer imagens de gatos. O ponto de partida podem ser os pesos de uma rede semelhante que foi construída e treinada para reconhecer imagens de cães. Desta forma, a rede precisará apenas se adaptar para reconhecer melhor as características desta outra classe.

Esta técnica pode ser usada mesmo que as classes não tenham tanta semelhança visualmente. Uma rede treinada do zero já teve que aprender, por exemplo, o que costuma fazer parte do plano de fundo e o que pode fazer parte de um objeto, bordas, texturas e assim por diante.

A transferência de conhecimento dá a possibilidade de redução no tempo de aprendizado e melhoria na performance final da rede.

3.1.2.7 Métricas de avaliação do aprendizado

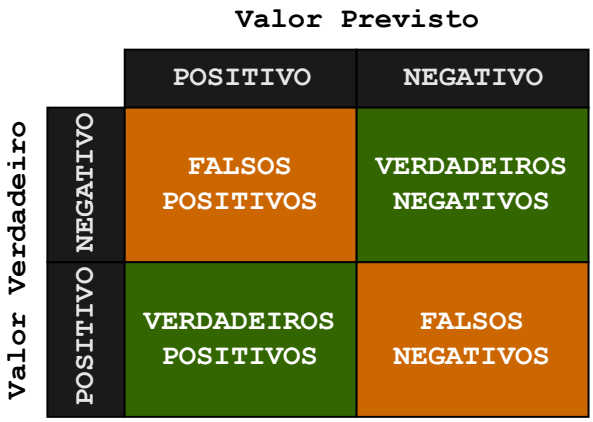
Existem vários métodos de se avaliar o desempenho de uma CNN, sendo que cada autor costuma escolher aquele que dê informações mais relevantes sobre sua aplicação.

No escopo da visão computacional, uma das métricas mais utilizadas pelos

autores é a *mean Average Precision* (mAP), a qual será apresentada neste capítulo. Contudo, primeiro serão apresentados conceitos que são pilares de sua composição, como a matriz de confusão, interseção sobre a união (do inglês *Intersection over Union* - IoU), precisão e *recall*.

Matriz de confusão – A matriz de confusão é um tipo de tabela que permite visualizar o desempenho de um algoritmo supervisionado através da apresentação das frequências da classificação para cada classe do problema. Para gerar a matriz de confusão, um conjunto de dados separados para o teste é classificado manualmente e, na sequência, aplicado à rede. Compara-se então os resultados previstos pela rede com os valores verdadeiros oriundos da classificação manual.

FIGURA 14 – MATRIZ DE CONFUSÃO



FONTE: O autor (2019)
LEGENDA: Matriz com valores previstos pelo modelo e os valores reais.

Como pode ser visto na FIGURA 14, são obtidos quatro conjuntos de dados. São eles: falsos positivos (FP) , quando a rede erroneamente identificou o dado como pertencente a uma classe da qual não era parte; verdadeiros positivos (VP), quando a rede corretamente identificou que o valor pertencia à determinada classe; verdadeiros negativos (VN), quando a rede corretamente identificou que um valor não pertencia à determinada classe e falsos negativos (FN), quando a erroneamente classificou o valor como não pertencente à determinada classe da qual fazia parte.

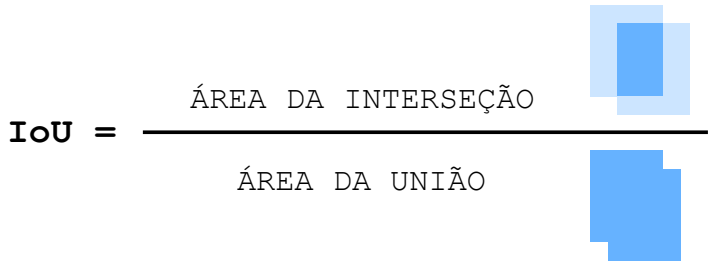
Suponha qua exista um banco de dados com imagens de dias ensolarados e dias nublados. Existem então quatro possibilidades de classificação: é dia ensolarado, não é dia ensolarado, é dia nublado e não é dia nublado.

Para uma imagem de dia ensolarado, se a rede a classificou como sendo dia ensolarado, então este valor é VP. Se classificou como dia nublado, este valor é FP. Se classificou como dia não ensolarado, o valor é FN. Se classificou como não nublado, o valor é VN. Este processo é realizado para todas as imagens do conjunto de teste e o total dos tipos de resultado é apresentado na forma da matriz de confusão.

Interseção sobre a União – Apresentada a matriz de confusão, há ainda mais um conceito importante na classificação de objetos em imagens. Frequentemente a tarefa que se realiza com a CNN não é apenas a de classificação (existência ou não de determinada classe na imagem), mas também de sua localização. Sendo assim, é necessário definir o que é considerado uma detecção correta.

No item 3.1.2.4 foi apresentado o conceito de caixa de seleção. Ao final do processo de inferência a CNN irá apresentar caixas de seleção que ela acredita pertencer a determinada classe. Cada caixa será comparada com a que foi manualmente rotulada. Como visto na FIGURA 15, será calculada a proporção conhecida como interseção sobre a união (IoU) dividindo-se a área da interseção entre a caixa prevista e a rotulada, com a união da caixa prevista com a rotulada.

FIGURA 15 – INTERSEÇÃO SOBRE A UNIÃO

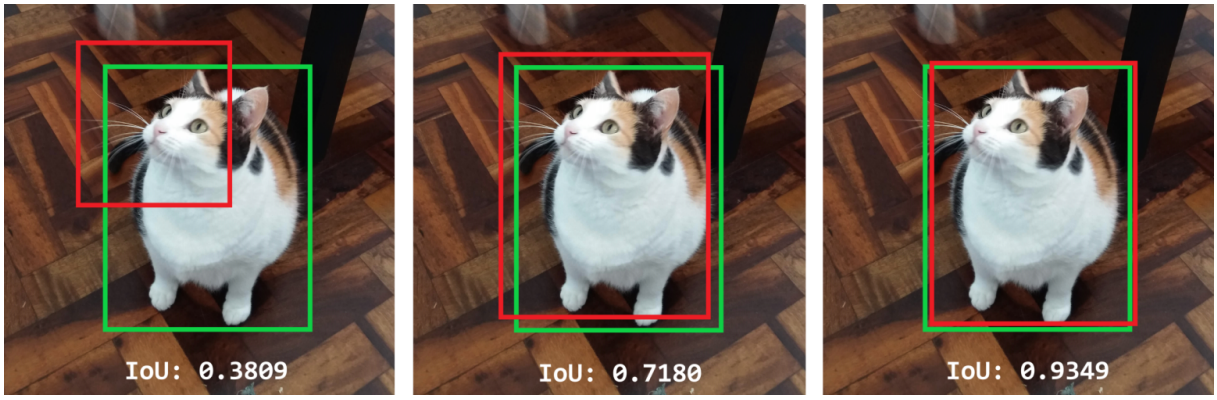


FONTE: O autor.
LEGENDA: Composição da proporção IoU.

Na FIGURA 16, em verde temos a caixa de seleção rotulada manualmente, conhecida também como *ground truth*, e em vermelho temos a caixa de seleção prevista pela rede. É possível observar que quanto mais próximas as duas caixas estão, mais perto de 1 é o resultado do IoU. Na literatura é comum definir que uma previsão com IoU superior ou igual a 0,5 é verdadeira.

Desta forma, um verdadeiro positivo é quando a classe inferida está correta e o IoU é maior ou igual a 0,5 (ou o valor definido pelo usuário como sendo limítrofe);

FIGURA 16 – IOU:EXEMPLO



FONTE: O autor.
NOTA: Os valores de IoU são aproximados, utilizados apenas pela didática.
LEGENDA: Exemplo de sobreposição das caixas de seleção e o IoU resultante.

um falso positivo é quando a caixa é verdadeira mas a classe inferida está incorreta e assim, por diante, de maneira análoga ao que foi apresentado para a matriz de confusão anteriormente.

Acurácia – Uma métrica comumente encontrada na literatura é a acurácia (ou *accuracy*, do original em inglês). Ela é definida pela soma dos verdadeiros positivos com os verdadeiros negativos, divididos pelo total de amostras. Sendo assim, ela é uma medida de como o modelo é capaz de prever corretamente detecções positivas.

$$\text{acurácia} = \frac{\text{VP} + \text{VN}}{\text{Total}} \tag{3.24}$$

Deve-se tomar cuidado com esta métrica, contudo. Suponha que exista um banco de dados que possua 1000 imagens de céluas. Entre elas estão classificadas células saudáveis e células cancerígenas. Porém, trata-se de um tipo célula cancerígena que costuma ocorrer 2 vezes a cada 1000 casos apenas. Suponha que a detecção resultante do modelo para este banco de dados tenha dado 1000 células saudáveis e nenhuma cancerígena. Sendo assim, teremos 2 falsos positivos e 998 verdadeiros positivos. Ao calcularmos a acurácia do modelo, teríamos $\text{acurácia} = (998 + 0)/1000 = 0,998$. Ou seja, o modelo tem uma acurácia de 99,8%. Contudo, para este exemplo, o ponto crítico é saber o quão bom ele é para detectar os casos de célula cancerígena. Sendo assim, esta acurácia de quase 100% passa a falsa impressão de que ele modela bem o problema.

Dessa forma, outros tipos de métricas também são importantes, uma vez que cada uma dirá algo sobre a capacidade do modelo.

Precisão – A precisão (ou *precision*, do original em inglês) pode ser vista como uma métrica que avalia quantas das predições positivas estão de fato corretas.

$$\text{precisão} = \frac{\text{VP}}{\text{VP} + \text{FP}} \tag{3.25}$$

Ou seja, o valor da precisão mostra paras as previsões positivas a proporção que é classificada de maneira equivocada pelo modelo.

Revocação – A revocação (ou *recall*, do original em inglês) pode ser vista como uma métrica que avalia quanto dos verdadeiros positivos foram identificados corretamente.

$$\text{revocação} = \frac{\text{VP}}{\text{VP} + \text{FN}} \tag{3.26}$$

Ou seja, o valor da revocação mostra paras as previsões positivas a proporção que é classificada como negativa de maneira equivocada pelo modelo.

É bastante sutil a diferença entre precisão e revocação, sendo que sempre há um compromisso entre as duas. Por exemplo, caso o objetivo seja melhorar a precisão, ou seja, reduzir os falsos positivos, é posível que em compensação o modelo fique inclinado a apresentar um número maior de falsos negativos, piorando a revocação.

Mean Average Precision (mAP) – Uma vez que as métricas de previsão e revocação acabam tendo a relação mencionada anteriormente, criou-se uma das métricas mais utilizadas para se avaliar a qualidade da rede neural em detectar e localizar objetos, a precisão média (ou do original em inglês, Average Precision - AP).

Em resumo, a AP computa o valor da precisão em função da revocação, sendo que este último irá variar de 0 a 1.

Para facilitar o entendimento, utilizaremos o exemplo de Jonathan Hui, disponível em seu sitio na internet[3]. Suponha um banco de dados de imagens e que nele aparecem no total 5 maçãs. Serão coletadas as predições feitas pelo modelo, tendo sido escolhido um IoU $\geq$ 0,5, e então elas serão rankeadas de acordo com o valor da confiança da precisão (ou seja, de acordo com o quanto o modelo acredita que aquele resultado previsto para a classe está correto). A TABELA 1 mostra esta etapa do processo.

[3] O exemplo original pode ser encontrado em: https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173. Acesso: outubro de 2019.

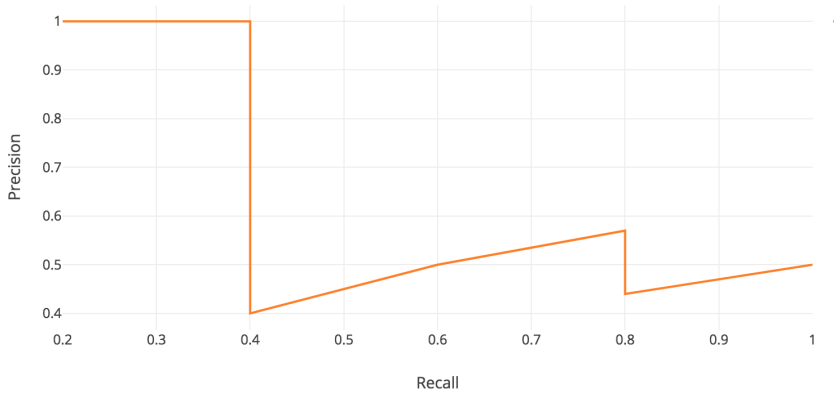
TABELA 1 – PRECISÃO MÉDIA (AP)

Rank	Correto?	Precisão	Revocação
1	Verdadeiro	1,0	0,2
2	Verdadeiro	1,0	0,4
3	Falso	0,67	0,4
4	Falso	0,5	0,4
5	Falso	0,4	0,4
6	Verdadeiro	0,5	0,6
7	Verdadeiro	0,57	0,8
8	Falso	0,5	0,8
9	Falso	0,44	0,8
10	Verdadeiro	0,5	1,0

FONTE: Adaptado de Jonathan Hui (2019).
LEGENDA: Exemplo de resultados inferidos por uma ANN.

Plotando então os valores da precisão pela revocação, temos como resultado o gráfico da FIGURA 17.

FIGURA 17 – CURVA DE MAP



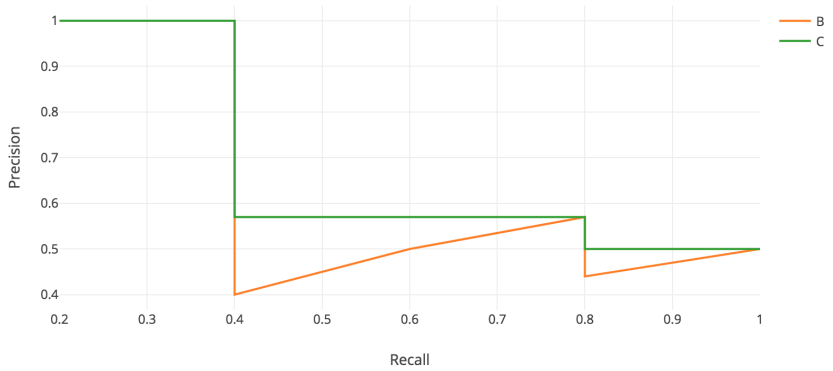
FONTE: Jonathan Hui (2019).
LEGENDA: Curva resultante do exemplo.

A precisão média, então, é definifa como a área sobre a curva da precisão em função da revocação:

$$AP = \int_0^1 p(r)dr \tag{3.27}$$

Contudo, antes de se calcular este valor, a curva é suavizada da maneira que pode ser vista na FIGURA 18 para a curva C, em verde.

FIGURA 18 – SUAVIZAÇÃO DA CURVA DE MAP



FONTE: Jonathan Hui (2019).
LEGENDA: Em laranja: curva original. Em verde: curva suavizada.

Como a revocação irá variar entre 0 e 1, como resultado teremos que a AP também irá variar entre 0 e 1.

Por fim, será calculada a média de todas as APs resultantes para cada classe presente no banco de dados. Isto é a métrica conhecida como mean Average Precision - mAP. Quanto mais próximo de 1 é este valor, melhor é a performace do modelo.

3.1.3 Camadas das Redes Neurais Convolucionais

As redes neurais convolucionais tem sua inspiração inicial no trabalho de Hubel e Wiesel (1959), o qual verificou que o cortex visual primário do cérebro dos gatos possuia neurônios organizados na forma de camadas. Estas, por sua vez, aprendiam a reconhecer padrões por primeiramente extrair características locais das imagens e subsequentemente combiná-las para obter representações de nível mais alto.

Este trabalho culminou no Neocognitron, proposto por Fukushima e Miyake (1982), que pode ser visto como uma forma inicial de CNN. Este trabalho consistiu na aplicação de múltiplas camadas que automaticamente aprenderam uma hierarquia de abstração de características para o reconhecimento de padrões.

Antes das CNNs, a parte de extração de características dos dados era feita de maneira manual e depois apresentada à rede neural artificial. Por exemplo, suponha que haja um banco de dados com imagens de dois tipos de flores e que deseja-se criar uma rede neural para classificá-las como pertencentes à primeira classe ou à segunda classe. Seria necessário anotar para cada imagem do banco de dados as características que o desenvolvedor julgasse discriminantes, como largura e comprimento das pétalas, por exemplo (entre outras possíveis), e então apresentar estes dados à rede para treinamento. Este processo é trabalhoso e nem sempre as características escolhidas pelo desenvolvedor são as mais relevantes de fato para

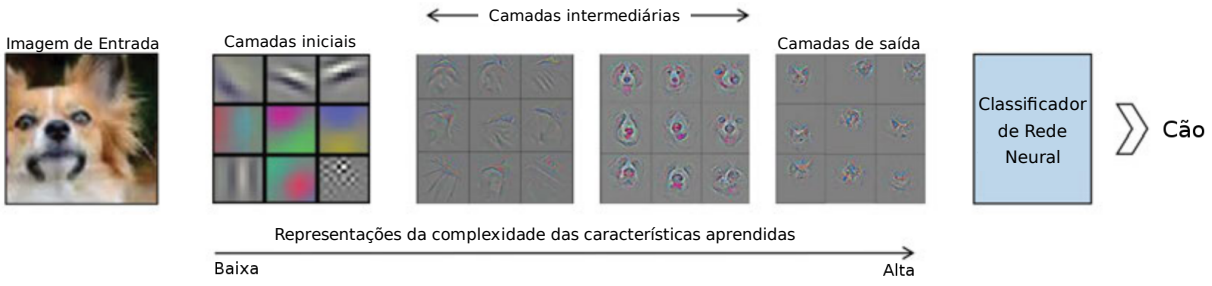
a rede, que não tem a mesma compreensão que um ser humano. Já no caso das CNNs isto é feito de maneira automática, ou seja, a rede dicidirá quais características são relevantes, sendo elas compreensíveis para um ser-humano ou não. A extração de características pode ser vista quase como uma caixa preta. Embora em alguns tipos de problema ainda seja necessário rotular dados para mostrar à rede em qual região está a informação de interesse, fica suprimido o processo trabalhoso da extração manual de características.

Desde de a criação do Neocognitron as CNNs se desenvolveram e ganharam grande notoriedade, especialmente em aplicações que envolvem dados com alta dimensionalidade como imagens e vídeos. Isto se dá principalmente após Y. LeCun et al. (1989) terem aplicado com sucesso o algoritmo de retropropagação para o processo de aprendizagem de uma rede neural construída para a tarefa de detecção de digitos escritos à mão.

Segundo a definição de Khan et al. (2018), o que distingue uma rede neural convolucional de uma rede neural comum é que cada unidade em uma CNN é um filtro, o qual é convolvido através das entradas das camadas. Estes filtros, por sua vez, incorporam o contexto espacial da midia de entrada da rede, porém em menor escala e mantendo a mesma forma espacial, de modo a representá-la, reduzindo a quantidade de variáveis que a rede precisa aprender.

A FIGURA 19 exemplifica este processo, onde a entrada da rede é uma imagem digital, que é uma matriz de dados em que cada elemento (ou pixel) é uma função que descreve cor e brilho (GONZALEZ; WOODS, 2000), a qual tem mapas de diferentes graus de complexidade de suas características espaciais extraídas através da convolução dos filtros aplicados a cada camada, os quais passarão por uma rede neural que irá realizar a classificação dos dados.

FIGURA 19 – EXTRAÇÃO DE CARACTERÍSTICAS



FONTE: Adaptado de Khan et al. (2018).
LEGENDA: Extração de características realizada por uma CNN.

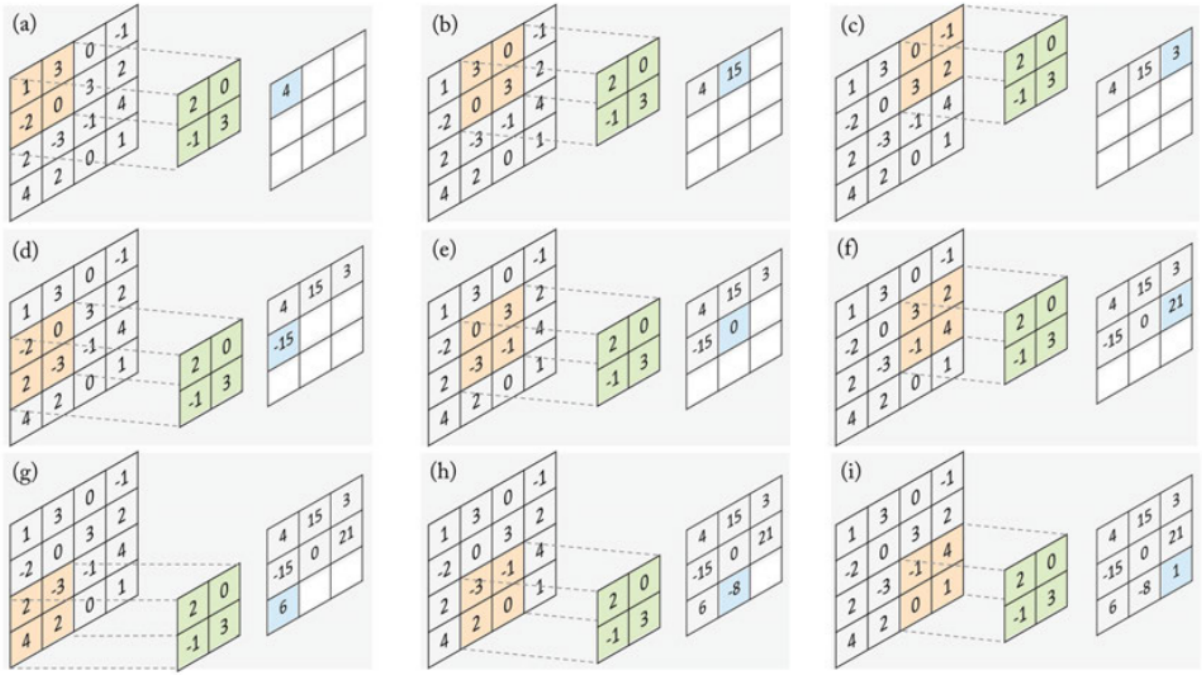
A seguir, neste capítulo, serão apresentados os blocos básicos que compõem a construção das camadas das redes neurais convolucionais.

3.1.3.1 Convolução

A camada de convolução, a partir da qual o nome das CNNs é atribuído, é uma camada composta por um conjunto de filtros[4] que serão convolvidos através de uma entrada, gerando em sua saída um mapa de características.

A FIGURA 20 exemplifica o processo de convolução[5]. Em verde temos o filtro, que é uma matriz de valores discretos de tamanho 2x2. Este filtro será convolvido, ou seja, "deslizado", através da camada de convolução, de tamanho 4x4, que é o mapa de características de uma entrada de duas dimensões. Os valores do filtro serão multiplicados, elemento a elemento, com os valores da camada de convolução sobrepostos por ele. Ao final, soma-se todos os elementos resultantes, o qual irá compor o mapa de característas de saída deste processo.

FIGURA 20 – CAMADA DE CONVOLUÇÃO



FONTE: Khan et al. (2018)
LEGENDA: Em laranja: valores originais. Em verde: Filtro. Em azul: valores resultantes da operação de convolução.

Neste exemplo o filtro dá um passo (ou *stride*, em inglês) unitário. Vide como o filtro, em verde, se move da FIGURA 20 (a) para FIGURA 20 (b). Mas este valor pode ser alterado. Suponha, por exemplo, um passo de tamanho 2. O filtro seria inicializado como na FIGURA 20

[4] Também chamados por alguns autores de kernels convolucionais.
[5] O processo exemplificado é, na verdade, a correlação cruzada e não a convolução, de acordo com a literatura de processamento de sinais. Para ser uma convolução o filtro primeiramente seria invertido verticalmente e depois deslizado pela camada. Contudo, em aprendizado de máquina, os dois termos costumam aparecer indistintamente.

(a) e, na sequência, passaria direto para o estado da FIGURA 20 (c), FIGURA 20 (g) e, por fim, FIGURA 20 (i). Ao deslizar por toda a camada, o mapa de características resultante seria uma matriz 2x2, e não 3x3, como é o caso apresentado no exemplo para um passo de tamanho unitário.

Sendo assim, é possível observar que a dimensão do mapa de características é reduzido mantendo de certa forma a distribuição espacial dos dados. Este processo recebe o nome de operação de subamostragem (ou *sub-sampling*, do inglês). Segundo Khan et al. (2018), este processo oferece uma moderada invariância a escala e posição dos objetos, o que é uma propriedade útil em aplicações como a de reconhecimento de objetos.

Existem ainda outras maneiras de configurar o filtro, seja em dimensão ou configuração dos seus valores, para que sejam capazes de extrair características dos dados que sejam mais adequados ao tipo de aplicação, que pode ser a remoção de ruídos em imagens, segmentação, super-resolução, e assim por diante.

3.1.3.2 Pooling

Outra camada importante em uma CNN é a camada de *pooling*. Nesta camada, blocos de valores são combinados através de funções como média e máximo. Assim como no caso do filtro da camada de convolução, também se define o tamanho do bloco e do passo no qual a função de *pooling* será aplicada.

A FIGURA 21 exemplifica este processo. Foi definido um bloco de tamanho 2x2 e um passo de tamanho unitário. Aplica-se, então, aos valores do mapa de características na entrada da camada de *pooling* a função máximo. Na saída da camada temos como resultado um mapa reduzido, no qual foi computado os valores máximos de cada bloco, no qual se concentra a característica com mais peso de cada região.

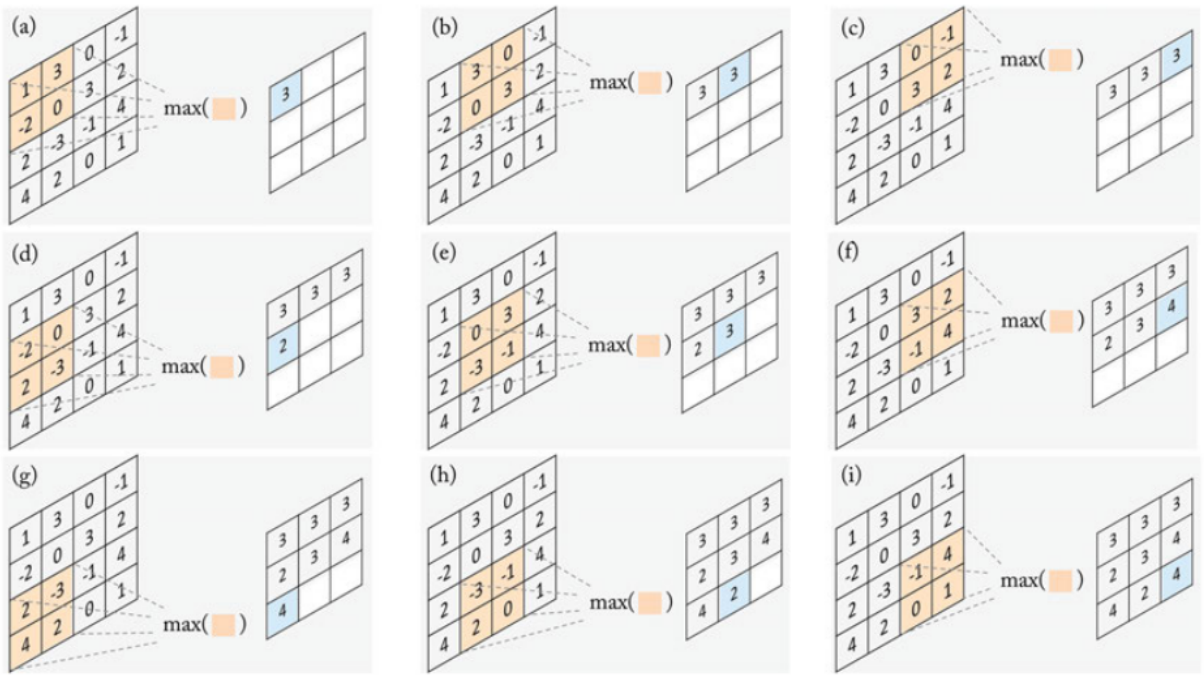
3.1.3.3 Camada totalmente conectada

Uma camada camada totalmente conectada (*fully connected layer*, em inglês) é essencialmente uma camada de convolução com filtros de tamanho 1x1, na qual cada unidade desta camada é densamente conectada à camada anterior (KHAN et al., 2018). Esta camada geralmente se localiza nos últimos estágios de uma CNN. Ela pode ser definida como:

$$\mathbf{y} = f(\mathbf{W}^T\mathbf{x} + \mathbf{b}) \tag{3.28}$$

onde $f(.)$ é uma função não linear, $\mathbf{x}$ é o vetor das entradas, $\mathbf{y}$ é o vetor das saídas, $\mathbf{W}$ é a matriz dos pesos das conexões entre os neurônios e $\mathbf{b}$ o vetor dos valores dos limiares. Ou seja, esta camada é análoga ao que foi visto para o MLP.

FIGURA 21 – CAMADA DE POOLING



FONTE: Khan et al. (2018)
LEGENDA: Operação de pooling. Em laranja: valores originais. Em azul: valor máximo da região.

3.1.3.4 Funções de ativação

As funções de ativação dos neurônios é o modo pelo qual é possível inserir não-linearidade à rede, permitindo que eles se comportem de maneira mais próxima ao funcionamento do neurônio biológico. As funções de ativação tomam um valor de entrada real e o comprime em um intervalo geralmente como [0, 1] ou [-1, 1].

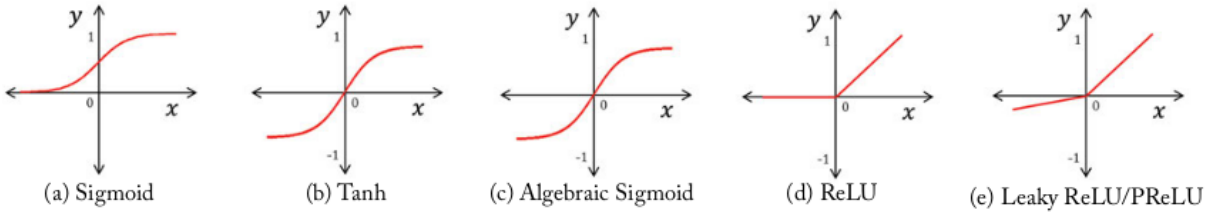
Uma função não linear pode ser entendida como um mecanismo de comutação ou de seleção, o qual irá decidir se um neurônio se ativa ou não dados todos os seus valores de entrada (KHAN et al., 2018).

A seguir serão apresentadas as funções de ativação que aparecem com mais frequência no escopo das ANNs.

Sigmoid – A função de ativação sigmoidal (FIGURA 22 a) irá receber um valor real em sua entrada e apresentar uma saída no intervalo [0, 1].

$$f_{sigm}(x) = \frac{1}{1 + e^{-x}} \tag{3.29}$$

FIGURA 22 – FUNÇÕES DE ATIVAÇÃO



FONTE: Adaptado de Khan et al. (2018).
LEGENDA: Funções responsáveis por adicionar não-linearidade à ANN.

Tanh – A função de ativação tangente hiperbólica (FIGURA 22 b) é muito semelhante à função sigmoidal, ela irá receber um valor real em sua entrada porém apresentar uma saída no intervalo [1, -1].

$$f_{tanh}(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.30}$$

Algebraic Sigmoid Function – A função de ativação sigmoidal algébrica (FIGURA 22 c) é muito semelhante à função tangente hiperbólica, sendo que sua diferença está apenas no grau de inclinação da curva.

$$f_{a-sig}(x) = \frac{x}{\sqrt{1+x^2}} \tag{3.31}$$

Rectifier Linear Unit (ReLu) – Baseada no processamento do cortex visual humano (HAHNIOSER et al., 2000), a ReLu (FIGURA 22 d) é uma das funções de ativação com maior importância prática. Ela mapeia o valor de entrada do neurônio para 0, caso este valor seja negativo, e matém o valor original, caso este seja positivo. Dada esta característica, a ReLu é computada em grande velocidade.

$$f_{relu}(x) = max(0, x) \tag{3.32}$$

Leaky ReLu – Dado o sucesso da ReLu, muitas variantes surgem dela. Um exemplo é a Leaky Relu (FIGURA 22 d), que mantém o sinal de saída caso a entrada seja positiva e reduz a escala do valor caso ele seja negativo.

$$f_{l-rel}(x) = \begin{cases} x & \text{se} \quad x > 0 \\ cx & \text{se } x \leq 0 \end{cases} \tag{3.33}$$

3.1.3.5 Exemplo: MNIST

Apresentadas as principais características e elementos que compõem uma CNN, neste capítulo será apresentado um exemplo de aplicação que permite com que estes conceitos sejam entrelaçados.

Um dos bancos de dados de imagens mais populares atualmente é o MNIST database (na qual a sigla MNIST significa Modified National Institute of Standards and Technology). Ele é composto por imagens de dígitos de 0 a 9, tendo sido escritos a mão, sendo 60 mil imagens destinadas para o treinamento e 10 mil imagens destinadas para o teste. Estas imagens tem a dimensão de 28x28 pixels e estão em escala de cor preto e branco. Na FIGURA 23 observa-se um exemplo de como são os dados que compõem o MNIST.

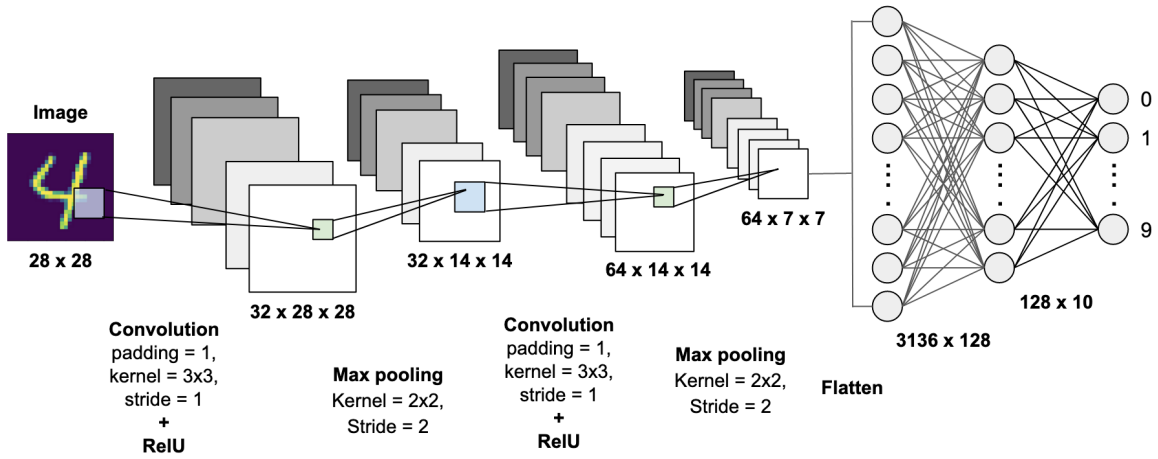
FIGURA 23 – MNIST

FONTE: Josef Steppan (2017).
LEGENDA: Exemplo dos dados de composição do MNIST.

A seguir, na FIGURA 23, está um exemplo de como uma CNN pode ser confeccionada para realizar a classificação destes dígitos. A rede apresentada tem caráter lúdico e, embora seja possível com este mesmo modelo classificar os dados, ela não necessariamente representa a melhor CNN já desenvolvida para esta tarefa.

Como é possível observar, na entrada da rede temos uma imagem de 28x28 pixels. Esta imagem terá suas características abstraídas através de uma camada de convolução, a qual é composta por um filtro de tamanho 3x3 e passo unitário. Além disso, é adicionada uma borda a imagem, de 1 pixel de dimensão, para que as representações resultantes mantenham o mesmo tamanho da imagem original. Nos neurônios desta camada é utilizada a função ReLu como ativação. Como resultados temos 32 camadas de saída, cada uma com 28x28 píxels, na quais estão representações espaciais da imagem que foram extraídas pela rede.

FIGURA 24 – CNN APLICADA AO MNIST



FONTE: Krut Patel (2019).
LEGENDA: Exemplo de CNN utilizada para classificar dados do MNIST.

Na sequência temos uma camada de pooling, com um filtro de tamanho 2x2 e passo de tamanho 2. Isto fará com que as camadas tenham seu tamanho reduzido pela metade.

Mais uma vez é aplicada a camada de convolução e pooling, idênticas às apresentadas anteriormente. Como resultado, temos 64 representações das características da imagem original, cada uma com 7x7 pixels.

Estes pixels são 'achatados', ou seja, dispostos em um único vetor de dados, contendo de maneira sequencial todos os pixels de todas as imagens. O resultado é um vetor de 3136 valores.

Estes valores serão aplicados à camada de entrada da ANN que é a parte da rede responsável por realizar a classificação dos dados resultantes da extração de características. Sendo assim, a entrada da rede é composta por 3136 neurônios. Na sequência é adicionada uma camada escondida para ajudar na tarefa de classificação e, por fim, uma camada de saída. Esta última contém 9 neurônios, uma vez que deseja-se classificar a imagem em sendo um valor entre 0 e 9. É importante ressaltar que não necessariamente estes valores precisam ser decimais, podendo ser, por exemplo, uma codificação do tipo one-hot.

Serão então realizadas várias épocas de treinamento, nas quais os pesos da rede neural e dos filtros das camadas de convolução e pooling serão calculados. Isto ocorrerá através dos algoritmos de retropropagação e método do gradiente.

Uma vez que foi observado que o erro apresentado pela rede tende a zero, ou seja, a função custo tendeu para um mínimo, utiliza-se o conjunto de dados de teste para avaliar a qualidade da rede na tarefa de classificação de dados.

3.2 CNNS UTILIZADAS EM DETECÇÃO DE OBJETOS

Existem várias arquiteturas de redes neurais convolucionais que se tornaram populares, principalmente no âmbito das tarefas de detecção e localização de objetos em imagens, nos últimos anos. Contudo, hoje em dia, os sistemas que apresentam melhores resultados costumam se utilizar de diversas técnicas em conjunto com as CNNs para melhorar a performance. Estes sistemas são conhecidos como detectores de objetos, no qual as CNNs clássicas e de estado da arte são utilizadas dentro do sistema para realizar principalmente a tarefa da extração de características das imagens. Neste caso, ela recebe o nome de *backbone*.

Existem dois tipos de detectores de objetos em imagens que se tornaram bastante populares nos últimos anos, segundo Zhao et al. (2019). O primeiro é o tipo dois-estágios, o qual costuma apresentar as melhores acurácias nas tarefas de detecção de objetos. Já os do segundo tipo, os de um-estágio, embora costumem apresentar precisão aquém da abordagem anterior, possuem uma maior simplicidade em sua arquitetura, o que contribui também para que sua performace seja superior no momento da inferência em termos de velocidade de processamento.

A seguir, além destes dois tipos de detectores serem aprofundados, também serão discutidas as arquiteturas mais populares e de estado da arte sobre redes neurais convolucionais utilizadas na tarefa de detecção de imagem.

3.2.1 Arquiteturas de CNNs

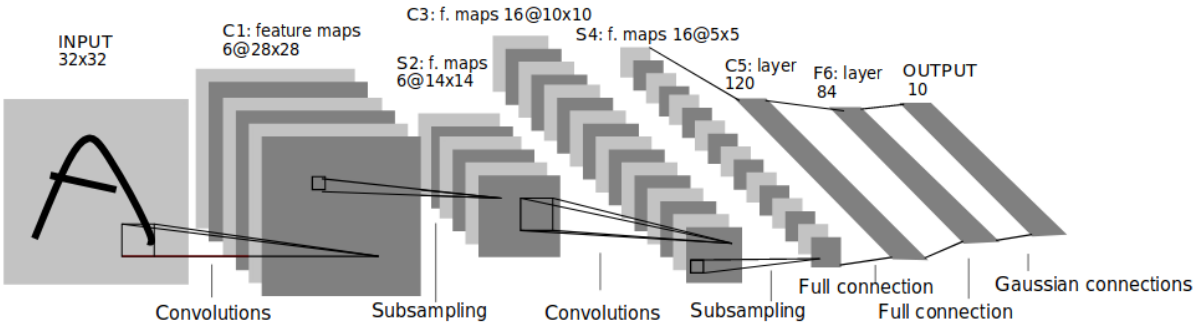
3.2.1.1 LeNet-5 (1998)

Em princípio, a extração de características das imagens era feita de maneira manual, uma tarefa despendiosa e demorada. Um marco que fez este tipo de tarefa evoluir das técnicas clássicas de aprendizado de máquina para o campo do aprendizado profundo foi o surgimento da LeNet-5 (Yann LECUN et al., 1998), uma rede neural convolucional utilizada para automaticamente classificar digitos escritos à mão. Para os padrões atuais, a LeNet-5 era uma rede simples, consistindo em apenas 7 camadas, como pode ser visto na FIGURA 25. São três camadas de convolução C1, C3 e C5, duas camadas de *pooling*, S1 e S4, uma camada totalmente conectada e uma camada de saída. Mesmo sendo simples, a LeNet-5 conseguiu atingir uma taxa de erro inferior a 1% no banco de dados MNIST.

3.2.1.2 AlexNet (2012)

Outro marco importante no desenvolvimento das CNNs foi a AlexNet (KRIZHEVSKY; SUTSKEVER; G. E. HINTON, 2012), desenvolvida para classificar as 1000 diferentes classes em 1.2 milhões de imagens do banco de dados ImageNet (DENG et al., 2009), atingindo uma taxa de erros de 7,5% para top-1 e 17,0% para o top-5 (que são métricas de performance definidas para desafio de classificação do ImageNet). Como pode ser visto na FIGURA 26, a

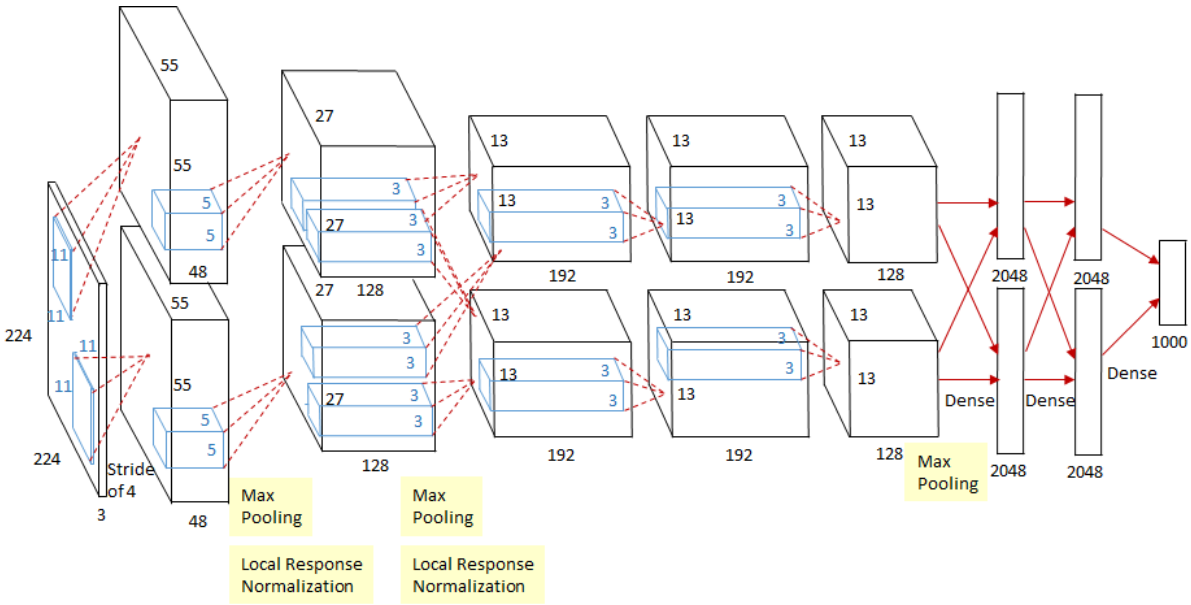
FIGURA 25 – LENET-5



FONTE: Yann LeCun et al. (1998).
LEGENDA: Arquitetura da CNN LeNet-5

AlexNet já trata-se de uma arquitetura bem mais complexa do que a LeNet-5. Aqui os dados estão sendo representandos pela suas dimensões, por exemplo, na camada de entrada temos uma imagem com resolução de 224x224 pixels, e 3 (camadas de cor) de profundidade. A partir da LeNet-5, métodos clássicos de aprendizado de máquina e de aprendizado profundo ainda coexistiam. Contudo, com a AlexNet obtendo uma performance excelente para imagens de cunho geral, não só para digitos em escala binária, as técnicas de detecção de objetos em imagens se voltaram quase que por completo para o aprendizado profundo.

FIGURA 26 – ALEXNET



FONTE: Sik-Ho Tsang (2018).
LEGENDA: Arquitetura da CNN AlexNet.

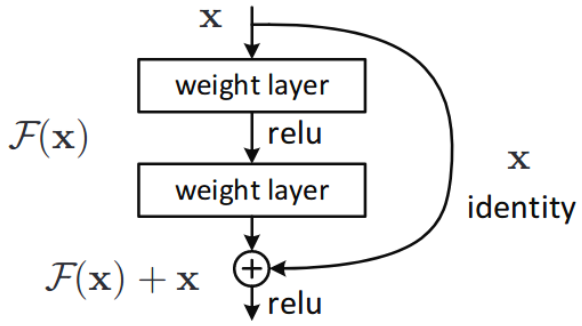
3.2.1.3 ResNet (2015)

Uma CNN que alcançou grande notoriedade desde sua criação e a rede de aprendizado residual profundo, a ResNet (HE et al., 2016). Ela surgiu através da observação de seus autores de que mais camadas de pronfundidade nas CNNs melhoravam os resultados obtidos em bancos de dados criados para competições sobre detecção de objetos em imagens, como o ImageNet (DENG et al., 2009) e o COCO (LIN; MAIRE et al., 2014), por exemplo. Sendo assim, eles resolveram averiguar até que ponto isto era verdadeiro, e notaram que após certo ponto a performance das redes saturava e os erros aumentavam. Além disso, há o problema da dissipação do gradiente, uma tendência do gradiente ficar cada vez menor conforme é retropropagado para o início da rede (NEAPOLITAN; NEAPOLITAN, 2018), fazendo a descida do gradiente se tornar insustentavelmente lenta.

Para contornar este tipo de problema, a ResNet utiliza um ”atalho” entre camadas, criando um caminho alternativo que permite que o gradiente seja retropropagado às camadas anteriores. Isto torna praticável o treinamento de redes mais profundas do que havia sido visto até então.

A FIGURA 27 mostra o bloco fundamental de construção da rede de aprendizado residual. É possível observar a existência de uma conexão de ”atalho”, a qual pode ou não ser ponderada, que faz uma ponte direta entre duas camadas que não estão dispostas de maneira subsequente.

FIGURA 27 – BLOCO DE ATALHO



FONTE: He et al. (2016)
LEGENDA: Composição da unidade fundamental da arquitetura ResNet.

Desta forma foi possível construir CNNs com profundidades nunca antes vistas. A TABELA 2 mostra as diversas profundidades nas quais a ResNet pode ser construída.

Embora criada em 2015, as ResNets ainda são um dos tipos de CNNs mais utilizadas como *backbone* em sistemas de detecção de objetos em imagens. Isto se dá uma vez que seus resultados continuam a se mostrar elevados e da extensa pesquisa realizada sobre elas, as comprovam sua robustes.

TABELA 2 – Composição da RESNET

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

FONTE: He et al. (2016).
LEGENDA: Configuração das camadas da ResNet dada sua profundidade.

3.2.1.4 Estado da Arte

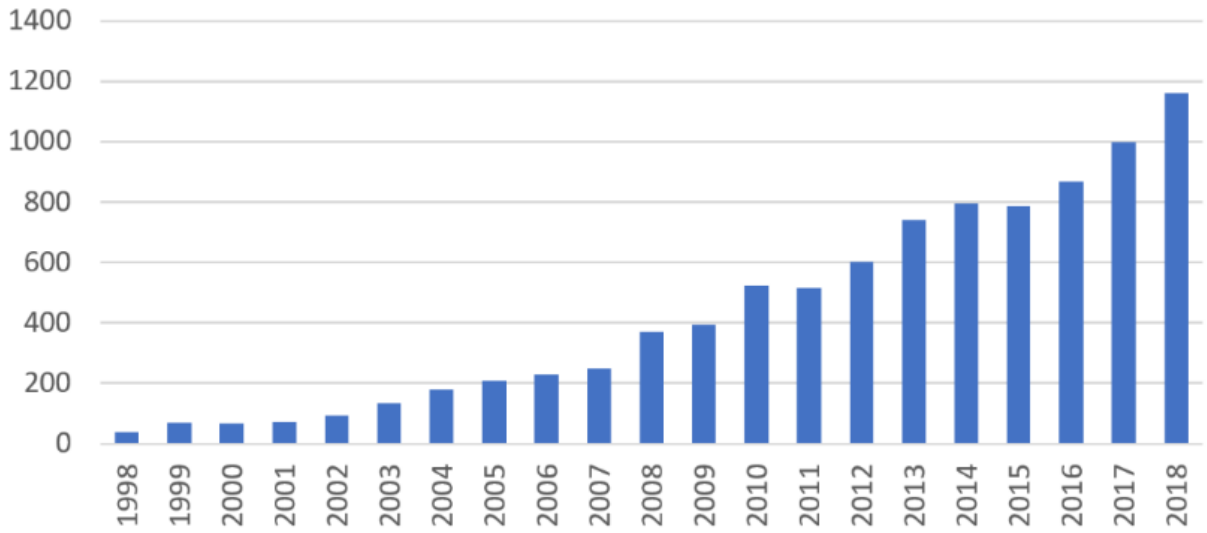
Além das arquiteturas mencionadas anterioremente, outras CNNs se desenvolveram de maneira paralela obtendo notoriedade, como a VGGNet (SIMONYAN; ZISSERMAN, 2014) e GoogLeNet (SZEGEDY et al., 2015), por exemplo.

Como este assunto tem sido extensivamente explorado tento pela academia quanto pela indústria, novas arquiteturas de CNNs surgem quase que diariamente. A FIGURA 28 mostra a evolução na quantidade de publicações sobre detecção de objetos nas últimas décadas.

Atualmente, entre as CNNs que representam o estado da arte na área de aprendizado profundo estão as redes generativas adversarias (do inglês *Generative Adversarial Network* - GAN), as quais conseguem gerar novos dados com estatísticas semelhantes aos dados de treinamento (GOODFELLOW et al., 2014), a DenseNet (HUANG et al., 2017) e ResNeXt (XIE et al., 2017), que podem ser vistas como uma evolução da ResNet e a SENet (HU; SHEN; SUN, 2018).

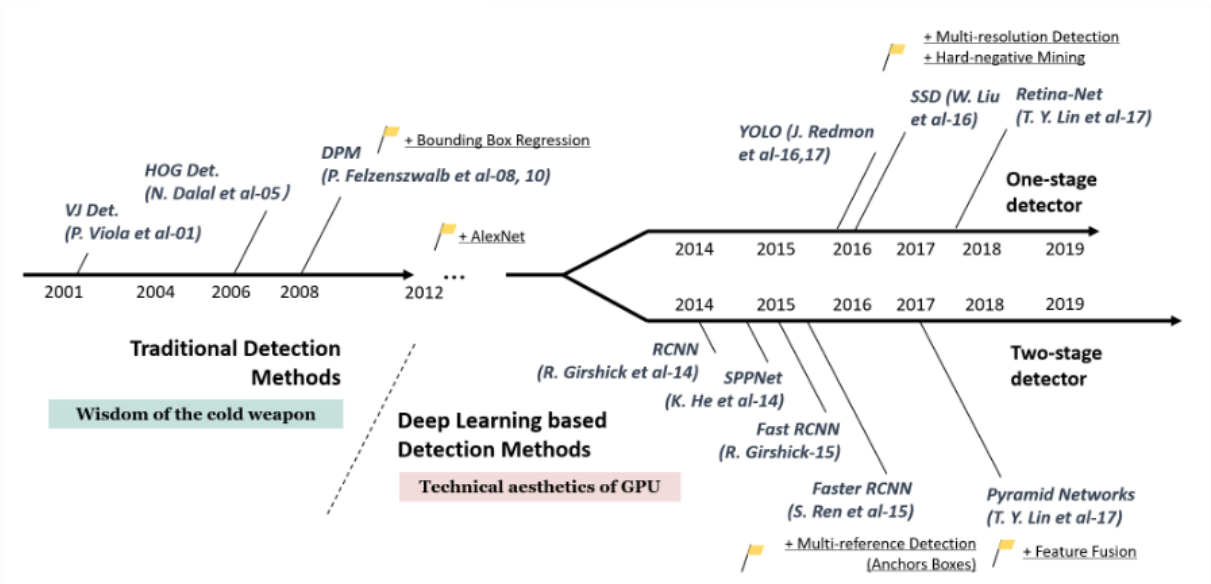
Na FIGURA 29 observamos mostra os marcos na evolução dos métodos de detecção de objetos em imagens. A linha do tempo começa na época em que eram utilizados métodos clássicos de aprendizado de máquina, o grande divisor de águas que foi o aprendizado profundo, e como as CNNs para detecção de objetos em imagens se dividiram em dois grandes grupos de abordagens. Estes grupos, dos detectores de objetos de um estágio e de dois estágios serão abordados nos items 3.2.2 e 3.2.3.

FIGURA 28 – PUBLICAÇÕES



FONTE: Zou et al. (2019).
LEGENDA: Evolução das publicações sobre detecção de objetos em imagens

FIGURA 29 – MARCOS



FONTE: Zou et al. (2019).
LEGENDA: Marcos na evolução da detecção de objetos em imagens.

3.2.2 Detectores de dois estágios baseados em CNNs

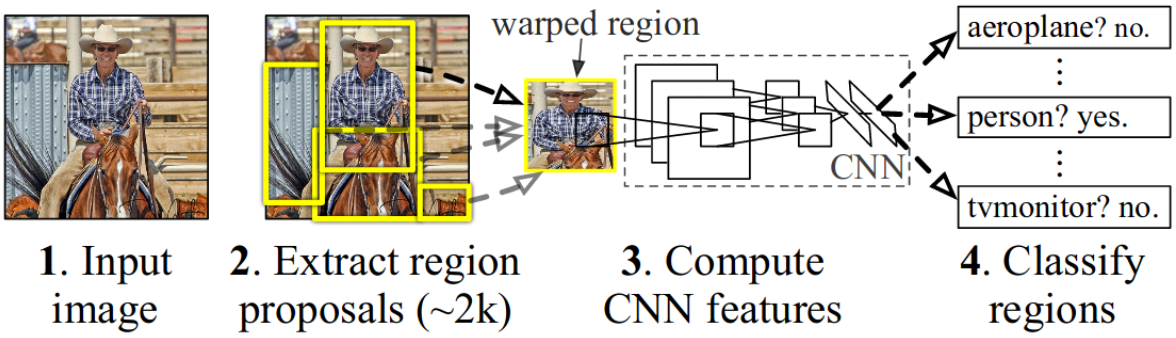
Os detectores de objetos de dois estágios recebem este nome pois sua arquitetura se divide em duas fases, sendo elas a de localização do objeto na imagem a fase subsequente de classificação. Nesta seção serão apresentados os detectores de dois estágios de estado da arte.

3.2.2.1 R-CNN

Um dos métodos mais importantes no escopo da detecção de objetos é o das CNNs baseadas em região (em inglês, *Region-based CNNs* - R-CNN), que surtiram com o trabalho de Girshick et al. (2014). A principal característica deste tipo de sistema é que em vez de se utilizarem do método tradicional de ”deslizar” um janela elas são capazes de extrair regiões de interesse através de uma busca seletiva.

Como pode ser visto na FIGURA 30, a R-CNN funciona da seguinte forma: primeiro, durante a chamada proposição de região (*region proposal*, do termo original em inglês) a rede extrai cerca de 2 mil áreas as quais ela julga relevantes. Após, as características de cada uma destas regiões são extraídas através de uma CNN (a AlexNet, no caso da R-CNN). Depois ocorre a classificação destas regiões.

FIGURA 30 – R-CNN



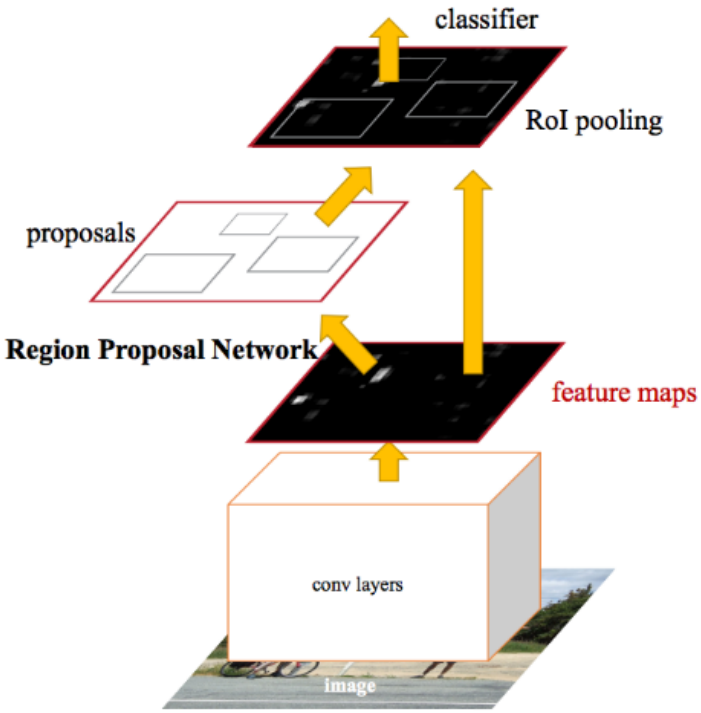
FONTE: Girshick et al. (2014).
LEGENDA: Arquitetura da R-CNN.

Por fim, há ainda mais um passo importante, um modelo de regressão linear é treinado sobre um das camadas de pooling, com o objetivo de gerar caixas de seleção em torno da localização dos objetos.

Naturalmente, várias caixas de seleção são propostas para cada objeto com diferentes probabilidades. Sendo assim, aplica-se uma técnica conhecida como *non-maximum suppression*, a qual irá se utilizar daquelas caixas de maior confiância para cada classe e analisar suas sobreposições, resultando em um caixa de seleção para cada objeto.

Este método evoluiu culminando nas redes Fast R-CNN (GIRSHICK, 2015), a qual passa utilizar Softmax para a etapa de classificação e Faster R-CNN (REN et al., 2017), a qual propõem um método conhecido como rede de proposição de regiões (em inglês *Region Proposal Network* - RPN). Como pode ser visto na FIGURA 31, a RPN está localizada logo após a extração de características pela CNN e, sendo assim, as mesmos mapas de características são utilizados para a classificação e proposição de regiões, o que impacta drásticamente no tempo de processamento e qualidade de inferência.

FIGURA 31 – FASTER R-CNN



FONTE: Ren et al. (2017).
LEGENDA: Arquitetuar da Faster R-CNN.

3.2.2.2 SCRDet

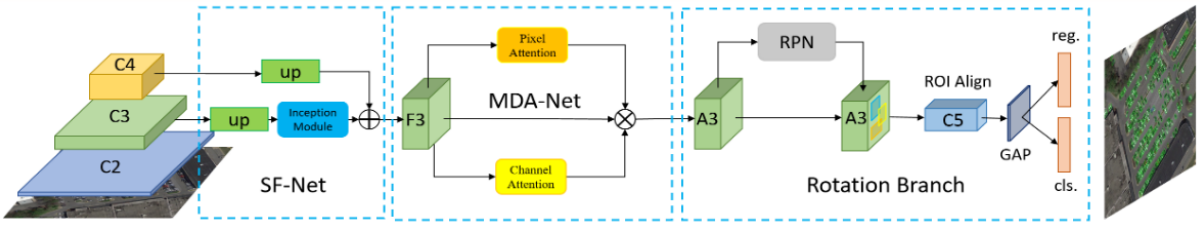
A SCRDet (anteriormente denominada R²CNN++) é um detector de objetos multi-classe, de propósito geral, criada com o objetivo de apresentar boa performance na tarefa de detecção de objetos em bancos de dados de imagens aéreas, como a DOTA (XIA et al., 2018) e o NWPU VHR-10 dataset (G. CHENG et al., 2014). Por esta razão, ela foi projetada para conseguir lidar com imagens que apresentem objetos de várias dimensões, em arranjos desordenados e densos, com orientações arbitrárias, e com planos de fundo ruidosos.

Além disso, o SCRDet também foi testado e mostrou bons resultados para detectar textos em cenas reais que apresentam variação de orientação, como do banco de dados ICDAR 2015 (KARATZAS et al., 2015).

Na FIGURA 32 vemos a composição da arquitetura da SCRDet, a qual pode ser dividida em três etapas principais, e que serão apresentadas a seguir.

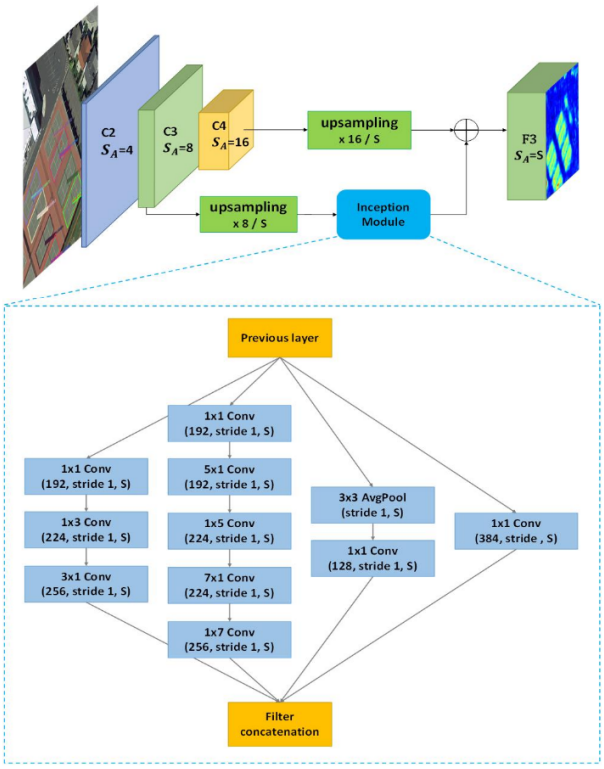
A primeira etapa da rede é a SF-Net, que consiste na utilização de camadas da ResNet (C1, C2, C3) em conjunto com outras camadas, para reduzir o conjunto de características que são relevantes para o problema.

FIGURA 32 – SCRDET



FONTE: Yang et al. (2019).
LEGENDA: Arquitetura da rede SCRDet.

FIGURA 33 – SFNET

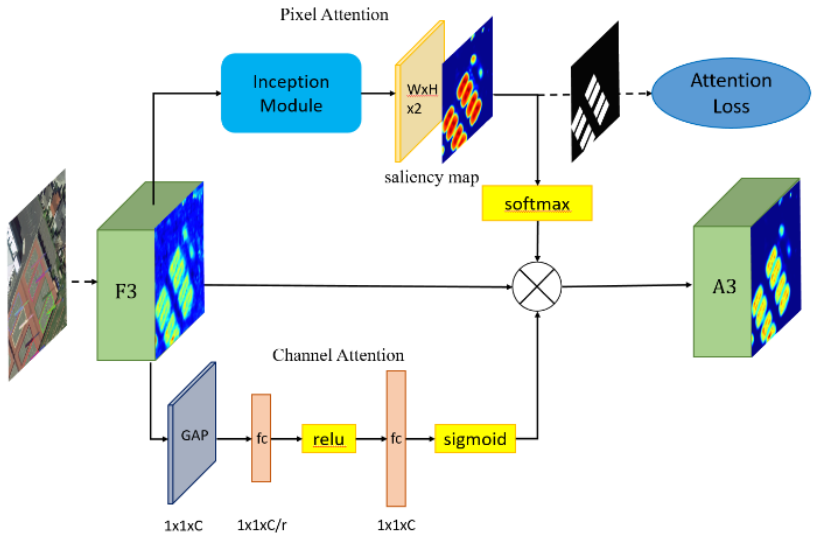


FONTE: Yang et al. (2019)
LEGENDA: Arquitetura do módulo SF-Net da rede SCRDet.

Na sequência está a MDA-Net (FIGURA 34), uma camada que tem como objetivo reduzir os ruídos e melhorar a separação entre a informação de interesse e aquela que pode prejudicar a qualidade da detecção.

Por fim, apresenta-se a etapa de rotação. Aqui a rede irá realizar a regressão das caixas de seleção, tanto orientadas quanto horizontais e realizar a classificação dos objetos

FIGURA 34 – MDA



FONTE: Yang et al. (2019)
LEGENDA: Arquitetura do módulo MDA-Net da rede SCR-Det.

encontrados.

A SCRDet ao longo de sua arquitetura faz uso de técnicas como RPN, de adição de outras pequenas redes convolucionais para melhorar os estágios, de camadas totalmente conectadas, a utilização de funções de ativação do tipo sigmoidal e ReLu, além da função de softmax.

A arquitetura é complexa, mas atualmente é uma das melhores redes para realizar a detecção de objetos que apresentam rotação.

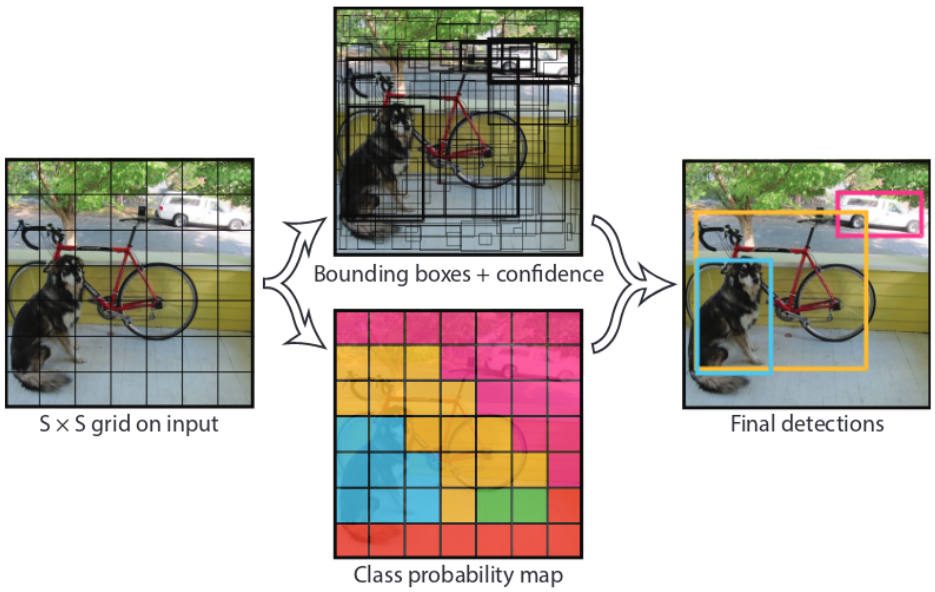
3.2.3 Detectores de um estágio baseados em CNNs

Ao contrário dos detectores de dois estágios, não há uma RPN nos detectores do tipo um estágio. Estes, por sua vez, tentam realizar tanto a classificação quanto a localização do objeto na imagem de uma só vez a partir do mapa de características.

Os dois detectores mais populares desta categoria são o You Only Look Once - YOLO (REDMON et al., 2016) e o Single Shot Detector - SSD (LIU et al., 2016).

A seguir, na FIGURA 35, observamos como este tipo de método funciona. A imagem de é tratada como uma grade de quadrados de mesmo tamanho e em cada uma destas partições a rede irá calcular a probabilidade de determinada classe estar ali. Por esta abordagem irá supor que o quadrado que apresenta a maior probabilidade para cada determinada classe é onde se encontra o centro daquele objeto. Sendo assim, a rede irá aplicar a técnica de *non-maximum supression* em todas as caixas de seleção no arredor desta partição retornando a área em que acredita estar o objeto.

FIGURA 35 – YOLO



FONTE: Redmon et al. (2016).
LEGENDA: Procedimento realizado por um detector de objetos de um estágio.

Em função de sua simpicidade, os detectores de um estágio acabam por ser mais velozes do que os do outro tipo, realizando a tarefa de detecção de objetos em um tempo expressivamente menor. Contudo, eles acabam sofrendo no quesito exatidão, uma vez que costumam se mostrar inferiores aos métodos com RPN.

Há, contudo, um sistema de um estágio que representa o estado da arte desta técnica, o qual tem se equiparado à exatidão de sistemas como Faster R-CNN, a RetinaNet (LIN; GOYAL et al., 2017).

3.2.3.1 RetinaNet

Detectores do tipo YOLO e SSD, embora tenham apresentado resultados promissores, ainda perdem no quesito acurácia em comparação a redes de dois estágios. Buscando encontrar em qual ponto as redes de um estágio deixam a desejar em comparação com as de dois estágios e afim de desenvolver um novo modelo que promova a melhoria na acurácia, Lin, Goyal et al. (2017) propõem a RetinaNet, uma rede que pela primeira vez se equiparou a performance média das redes de dois estágios em desafios clássicos de detecção de objetos, como o do banco de dados COCO (Lin, Maire et al. (2014)), por exemplo.

Os autores da RetinaNet identificaram como o maior problema para as redes de um estágio obterem performace do estado da arte era o desbalanceamento entre as classes durante o treinamento, ou seja, quando há uma quantidade expressiva de exemplos de uma classe a mais do que de outra.

No caso de detectores como R-CNN este problema é parcialmente endereçado, uma

vez que o estágio de proposição rapidamente reduz o número de localizações candidatas para o objeto, filtrando a maior parte das amostras de plano de fundo da imagem. Além disso, no segundo estágio de classificação, heurísticas de para a amostragem são aplicadas com o objetivo de manter um balanço entre as classes do primeiro plano e o plano de fundo.

Já as redes de um estágio, por sua arquitetura, acabam tendo que lidar com uma quantidade muito maior de proposições de localização de objetos e, embora existam métodos para tentar contornar o problema, eles são ineficientes. Isto ocorre uma vez que para o treinamento exemplos facilmente classificáveis de plano de fundo acabam por exercer domínio.

A solução encontrada pelos autores foi o desenvolvimento de uma nova função de custo inspirada na *cross entropy*, a qual recebeu o nome de *focal loss*. Esta função foca o treinamento em um conjunto esparso de exemplos difísseis de classificar, o qua previne a ocorrência de um vasto número de negativos facilmente classificável que acabam por sobrecarregar o detector durante o treinamento (LIN; GOYAL et al., 2017).

3.3 COMPARATIVOS

Embora as CNNs tenham sofrido grandes avanços na última década e serem possívelmente a melhor ferramenta de detecção de objetos em imagens da atualidade, estas ainda encontram dificuldades quando há grande variação de escala, orientação e forma, do objeto de interesse.

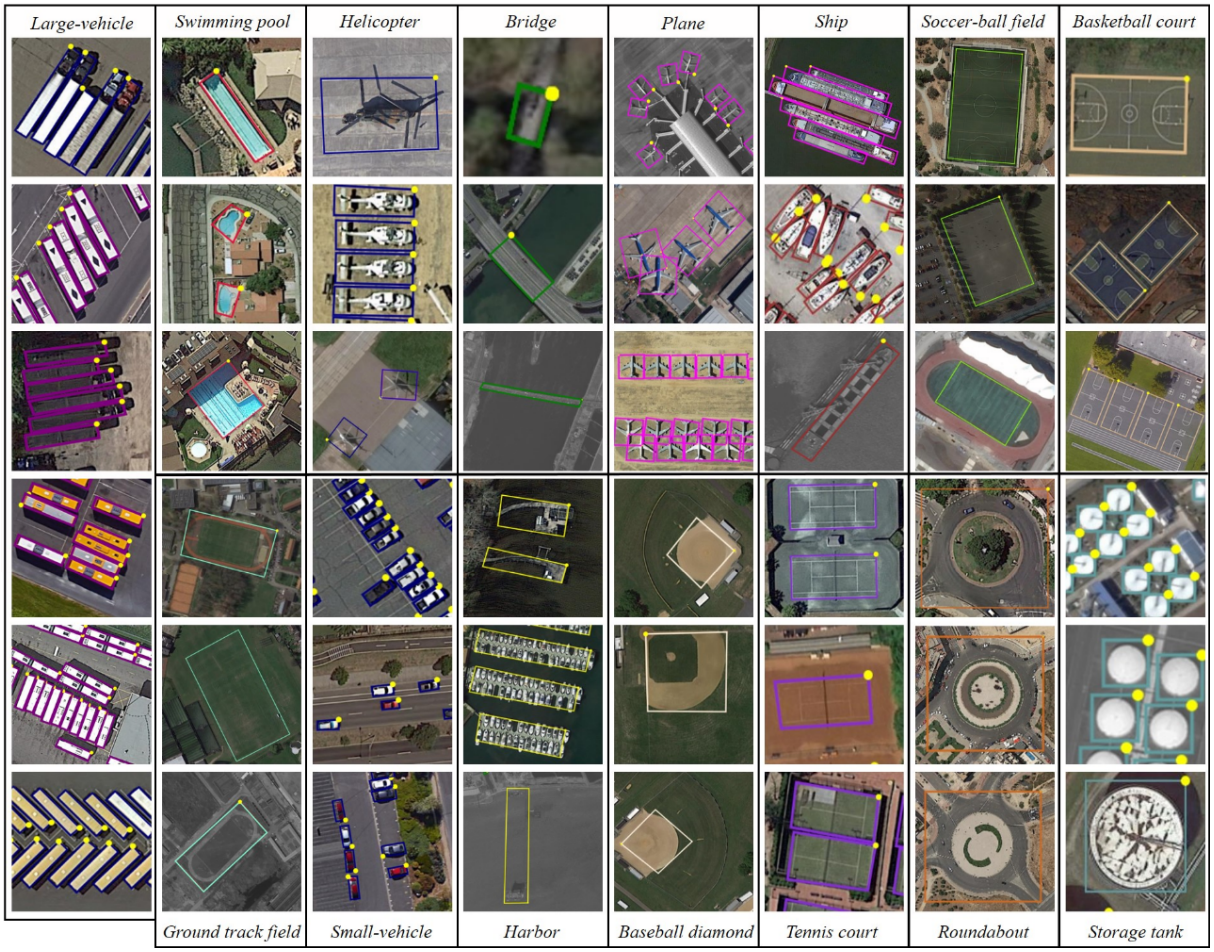
Um banco de dados, que em função de suas características, permite com que estes problemas citados anteriormente sejam abordados, e que ganhou rápida notoriedade é o Conjunto de Dados em Larga Escala para Detecção de Objetos em Imagens Aéreas (do original, em ingês, *A Large-scale Dataset for Object Detection in Aerial Images* - DOTA), o qual foi desenvolvido para a tarefa de detecção de objetos em imagens aéreas, e construído apartir de dados do Google Earth, e dos satélites JL-1e GF-2 do Centro de Recursos de Dados e Aplicação de Satélite da China.

São, no total, 2806 imagens com resoluções entre 800x800 a 4000 x 4000 pixels, contendo 15 classes de objetos. S]ao elas: avião, navio, tanque de armazenamento, camp de beisebol, quadra de tênis, quadra de basquete, pátio de trilhos, porto, ponte, veículo grande, veículo pequeno, helicóptero, rotatória, campo de futebol e piscina. Na FIGURA 36 temos um exemplo de como é a configuração do banco de dados DOTA.

Como é possível observar, as classes estão distribuídas de maneira a apresentar variação de escala, orientação e forma, com vários tipos de planos de fundo e composições.

A criação do DOTA culminou no desenvolvimento de diversos detectores de objetos que apresentam boa capacidade em classificar e localizar objetos com variabilidade em sua rotação.

FIGURA 36 – DOTA DATASET



FONTE: Xia et al. (2018).
LEGENDA: Exemplo de composição dos dados do DOTA.

A seguir, na TABELA 3, temos um comparativo da mAP (para um IoU $\geq$ 0,5) de vários sistemas detectores de objetos do tipo dois estágios, obtidos para a tarefa de detecção de objetos rotacionados no DOTA.

TABELA 3 – DETECTORES DE DOIS ESTÁGIOS - MAP@0,5

R-FCN	FR-H	FR-O	R-DFPN	R²CNN	RRPN	ICN	RoI-Trans	SCRDet
26,79	32,29	52,93	57,94	60,67	61,01	68,20	69,56	72,61

FONTE: Yang et al. (2019).
LEGENDA: Performance de detectores de dois estágios testados no DOTA.

A seguir, na TABELA 4, temos um comparativo da mAP (para um IoU $\leq$ 0,5) de vários sistemas detectores de objetos do tipo um estágio, obtidos para a tarefa de detecção de objetos rotacionados no DOTA.

TABELA 4 – DETECTORES DE UM ESTÁGIO - MAP@0,5

SSD	YOLOv2	R³Det+ResNet101	R³Det+ResNet152
17,84	25,49	71,69	72,81

FONTE: Xia et al. (2018)
LEGENDA: Performance de detectores de um estágio testados no DOTA.

Uma vez que este trabalho busca utilizar CNNs para detectar objetos com suas rotações (ângulo), os resultados obtidos pelas através do banco de dados DOTA deram suporte à escolha dos sistemas a serem implementados.

4 DESENVOLVIMENTO E RESULTADOS

Neste capítulo serão apresentados e avaliados os resultados obtidos ao longo da implementação do projeto. Este capítulo começa compilando as alterações que banco de dados sofreu ao longo do desenvolvimento.

4.1 BANCO DE DADOS DE POSTES E CHAVES

Não foi possível verificar, através do levantamento realizado, a existência de um banco de dados público contendo imagens dos objetos de interesse. Sendo assim, foi necessário que este projeto se iniciasse através da confecção do mesmo.

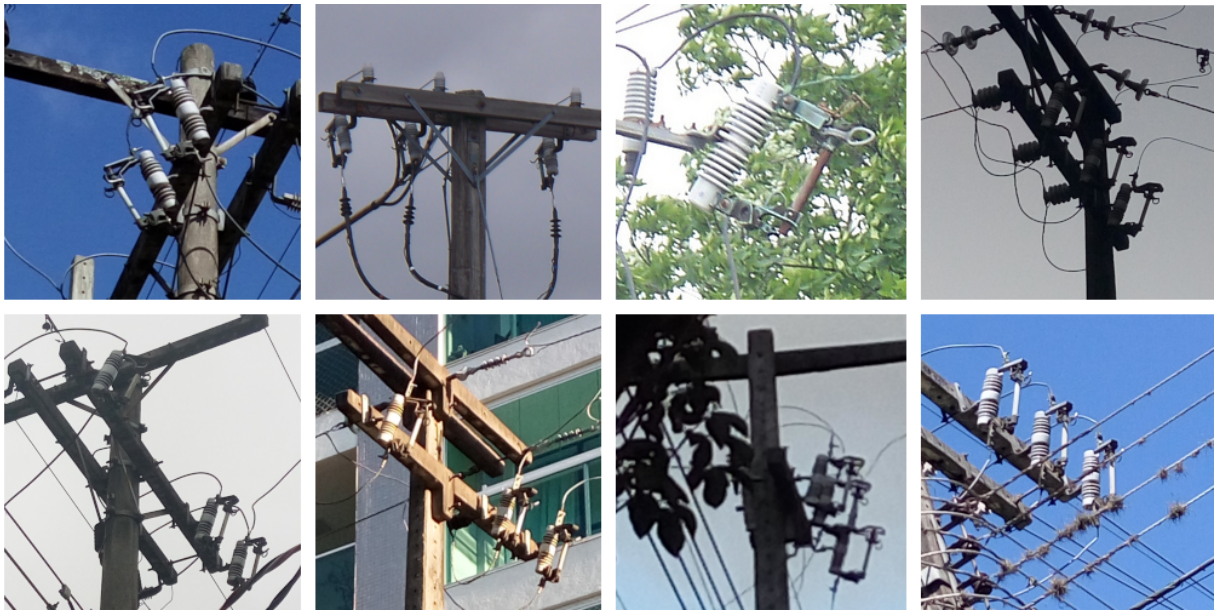
Para isto foram utilizadas imagens adquiridas através da internet, bem como de registros realizados em campo pelo próprio autor.

Nas figuras a seguir pode ser observado alguns exemplos das imagens que compõem este banco de dados, sendo elas referentes as classes ”poste” (FIGURA 37), ”chaves fechadas”(FIGURA 38) e ”chaves abertas”(FIGURA 39).

FIGURA 37 – POSTES

FONTE: O autor.
LEGENDA: Recortes de imagens retiradas do banco de dados de postes e chaves.

FIGURA 38 – CHAVES FECHADAS



FONTE: O autor.
LEGENDA: Recortes de imagens retiradas do banco de dados de postes e chaves.

FIGURA 39 – CHAVES ABERTAS

FONTE: O autor.
LEGENDA: Recortes de imagens retiradas do banco de dados de postes e chaves.

Buscou-se criar um banco de dados que apresenta-se uma boa diversidade de imagens, nas quais os objetos das classes de interessem estivesem inseridos em paisagens urbanas e rurais, em dias claros e nublados, diversos tipos de vegetação, ângulos e luminosi-

dade da fotogragia, e assim por diante.

As imagens possuem várias resoluções, sendo todas com o formato PNG e espaço de cores RGB.

Com o desenvolvimento da atividade, o banco de dados inicial (subseção 4.1.2) teve que ser ampliado (subseção 4.1.3) e posteriormente ajustado (subseção 4.1.4) para que se chegasse a um resultado final satisfatório.

4.1.1 Padronização da Anotação dos Dados

Para a notação dos dados optou-se por utilizar o padrão de anotação do PASCAL VOC, um dos bancos de dados mais populares em competições de detecção de objetos. Neste padrão, as informações presentes em cada imagem são anotadas em arquivos em formato XML (Extensible Markup Language), referentes a cada imagem. No ANEXO A pode ser visto um exemplo de anotação destes dados. Além disso, a FIGURA 40, mostra um exemplo de como são construídas as caixas de seleção em torno de cada objeto na imagem.

Por fim, estes dados acabam sendo convertidos para serem utilizados na rede SCRDet. O resultado desta conversão pode ser visto no ANEXO B.

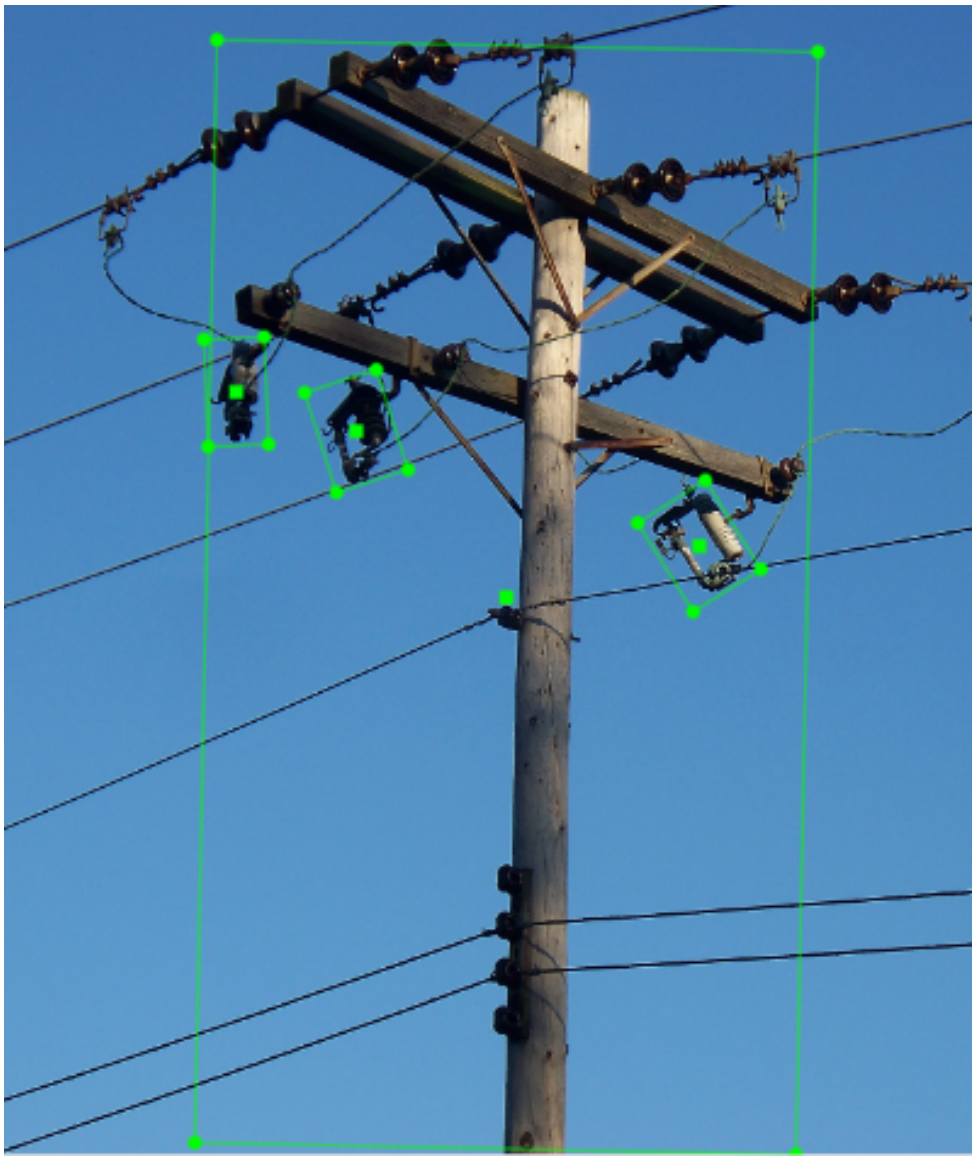
4.1.2 Inicial

O banco de dados inicial foi composto de um total de 426 imagens. A FIGURA 41 mostra a quantidade de objetos de cada classe que foram identificados nestas imagens. No total são 600 postes, 1143 chaves fechadas e 96 chaves abertas. Aí já ficou possível observar como esta última classe é bem mais rara de se verificar a ocorrência.

Como é importante conhecer os dados com os quais se está trabalhando, foi realizado um levantamento da frequência de ocorrência de cada uma das classes, para cada imagem, ao longo do banco de dados. Na FIGURA 42 vemos as distribuições dos postes em ciano, as chaves fechadas em magenta e as chaves abertas em amarelo.

Como era de se esperar, observa-se que as chaves abertas ocorrem de maneira mais esparsa ao longo do banco de dados.

FIGURA 40 – PADRONIZAÇÃO DAS ANOTAÇÕES



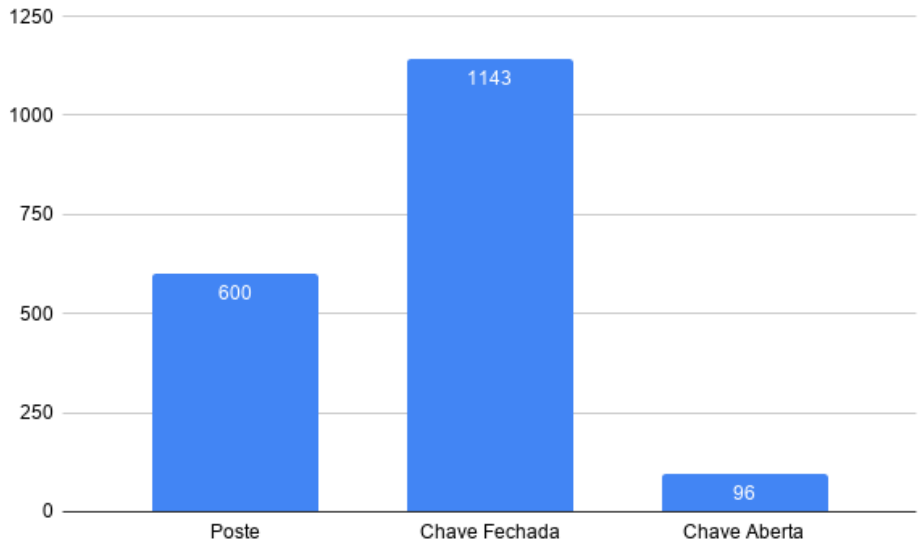
FONTE: O autor.
LEGENDA: Distribuição dos objetos por classe no banco de dados.

4.1.3 Ampliação

Este banco de dados foi ampliado seguindo o mesmo procedimento de sua confecção inicial, resultando agora em um total de 2252 imagens. Na FIGURA 43 vemos a nova configuração do banco de dados, em vermelho, em comparação com o bando de dados inicial, em azul.

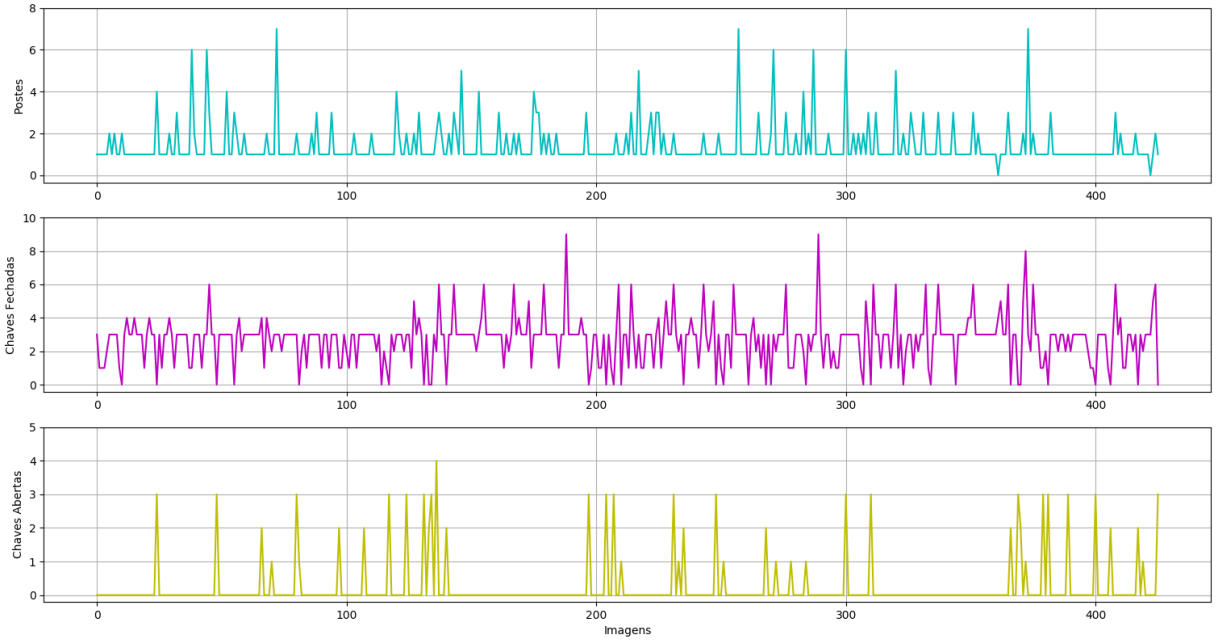
Mais uma vez nota-se a dificuldade em se coletar imagens de chaves em estado aberto.

FIGURA 41 – BANCO DE DADOS INICIAL



FONTE: O autor.
LEGENDA: Distribuição dos objetos por classe no banco de dados.

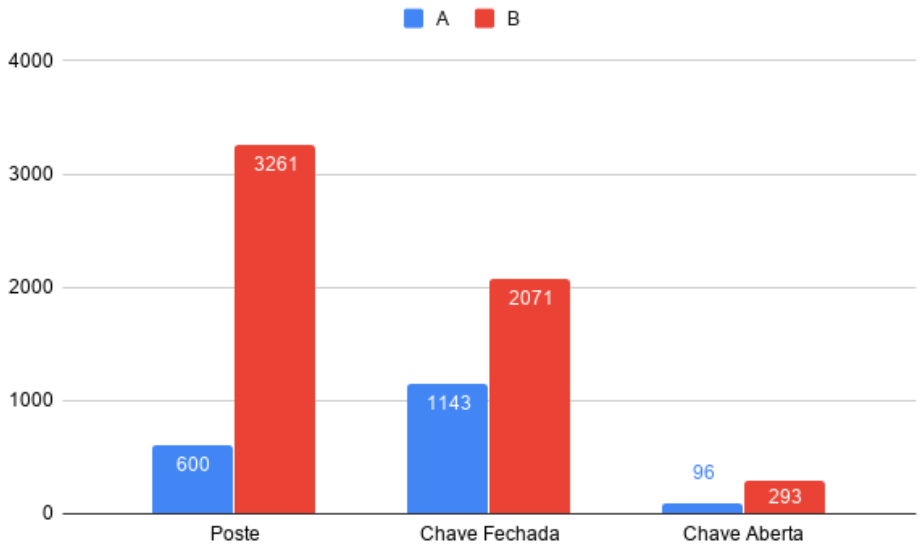
FIGURA 42 – BANCO DE DADOS INICIAL: DISTRIBUIÇÃO



FONTE: O Autor
LEGENDA: Frequência de ocorrência dos objetos por classe no bando de dados. Em ciano: postes. Em magenta: chaves fechadas. Em amarelo: chaves abertas.

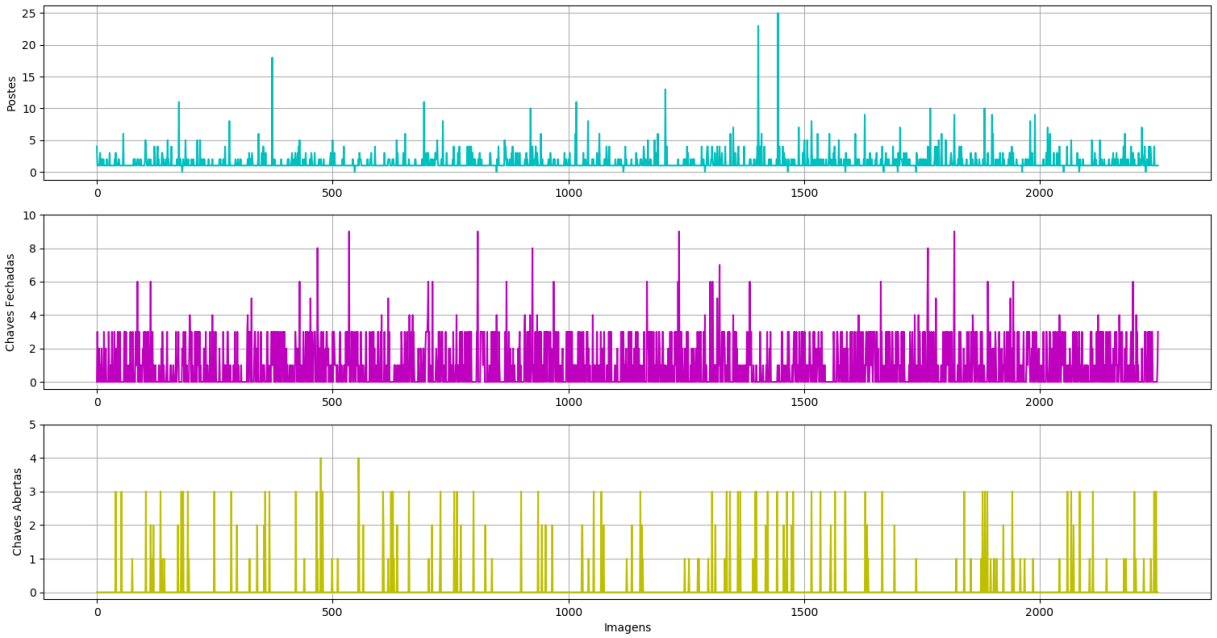
Na FIGURA 44, da mesma forma que na seção anterior, temos os dados de distribuição dos objetos de cada classe ao longo deste novo banco de dados.

FIGURA 43 – BANCO DE DADOS AMPLIADO



FONTE: O Autor
LEGENDA: Objetos por classe no bando de dados. Em vermelho: banco de dados ampliado. Em azul: banco de dados inicial.

FIGURA 44 – BANCO DE DADOS AMPLIADO: DISTRIBUIÇÃO



FONTE: O autor.
LEGENDA: Frequência de ocorrência dos objetos por classe no bando de dados. Em ciano: postes. Em magenta: chaves fechadas. Em amarelo: chaves abertas.

É possível observar como agora há uma melhor distribuição dos objetos ao longo do banco de dados.

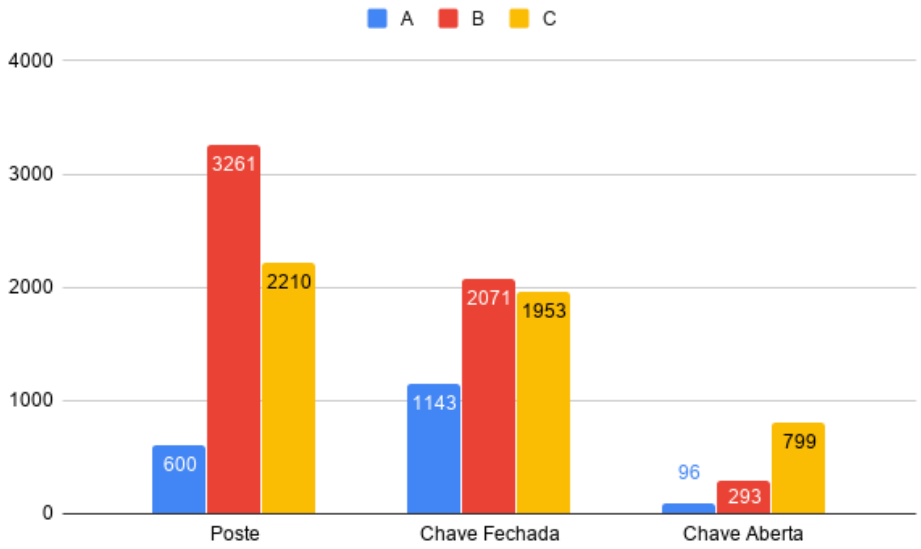
4.1.4 Aumento de Dados e Balanceamento de Classes

Ao longo do projeto verificou-se a necessidade da aplicação de técnicas de manipulação do banco de dados para a melhoria do modelo de detecção de postes e chaves.

Primeiramente aplicou-se a técnica de aumento de dados (subseção 3.1.2.5) para imagens que possuiam a classe ”chave aberta” e, na sequência, algumas imagens de postes foram deletadas. O objetivo desta etapa foi aumentar a variabilidade de exemplos de chaves abertas e proporcionar um balanceamento melhor entre as classes que compõem o banco de dados.

Na FIGURA 45 observamos a nova composição: em azul temos o banco de dados inicial, em vermelho o banco de dados ampliado e em amarelo o banco de dados após passar por esta etapa.

FIGURA 45 – BANCO DE DADOS COM AUMENTO DE DADOS E BALANCEAMENTO

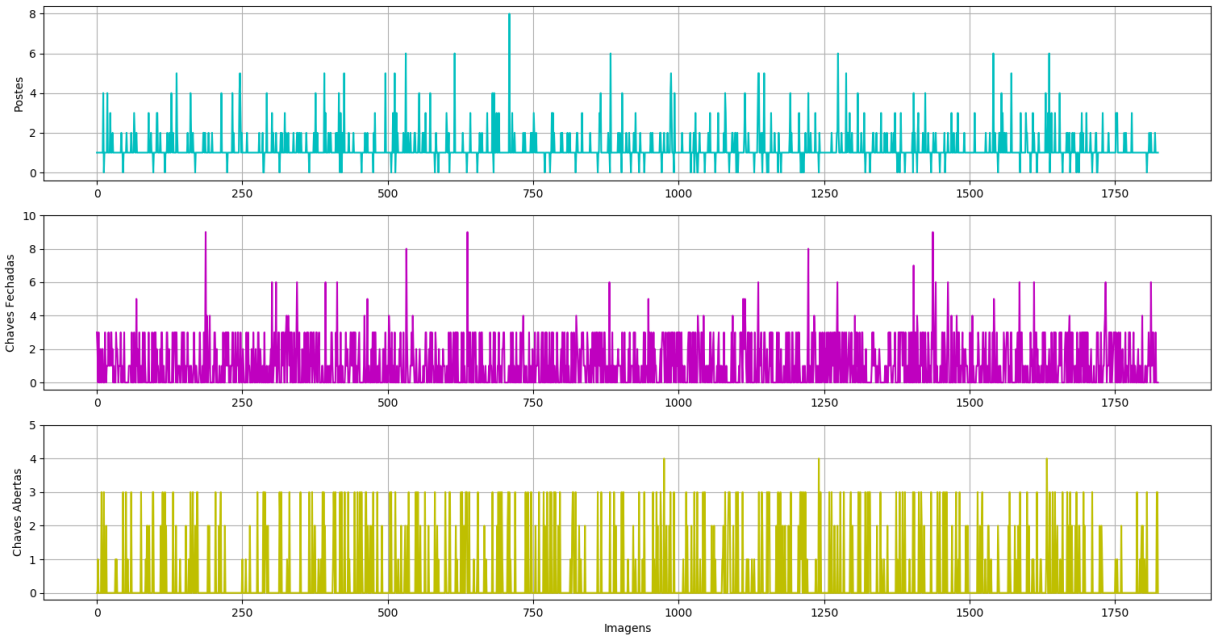


FONTE: O autor
LEGENDA: Objetos por classe no bando de dados. Em amarelo: banco de dados com aumento de dados e balanceamento. Em vermelho: banco de dados ampliado. Em azul: banco de dados inicial.

É importante ressaltar que embora seja possível até certo ponto extrapolar os exemplos através da técnica de aumento de dados, para obter um modelo mais confiável é necessário ter exemplos reais. Por esta razão, optou-se por ampliar a quantidade de representações de chaves abertas, mas não ao ponto em que seu valor se equiparasse à quantidade de postes e chaves abertas, que estavam muito distantes.

Por fim, na FIGURA 46 temos a distribuição dos objetos por classe ao longo do banco de dados.

FIGURA 46 – BANCO DE DADOS COM AUMENTO DE DADOS E BALANCEAMENTO: DISTRIBUIÇÃO



FONTE: O autor
LEGENDA: Frequência de ocorrência dos objetos por classe no bando de dados. Em ciano: postes. Em magenta: chaves fechadas. Em amarelo: chaves abertas.

4.2 CONFIGURAÇÕES DO AMBIENTE DE DESENVOLVIMENTO

Para a realização das atividades foi utilizado um microcomputador com processador Intel Core i7-9700k, com 32GB de memória RAM e uma placa de vídeo GeForce RTX 2080 Ti.

Foi utilizada a linguagem de programação Python (versão 2.x e versão 3.x) com as bibliotecas OpenCV, para processamento de imagens, e TensorFlow versão 1.4.1, para a implementação das redes neurais, entre outras. Foi utilizada também a plataforma voltada para ciência de dados Anaconda 2 e 3.

Foi utilizada também a API de processamento paralelo CUDA, da NVIDIA, na versão 8.0, rodando no sistema operacional Linux, na distribuição Ubuntu 18.04.

4.3 SCRDET

A implementação da SCRDet para a detecção de postes e chaves utilizou o código fonte da rede disponibilizado no repositório[1] de seus autores. Foram realizadas modificações para adequá-la quanto à dimensão e configuração dos dados, quantidade de classes, limitações computacionais e assim por diante.

[1] https://github.com/DetectionTeamUCAS/R2CNN-Plus-Plus_Tensorflow

Para encontrar a taxa de aprendizado mais adequada ao problema foram realizados testes que consistiam em aplicar um determinado valor e observar se a curva de treinamento estava tendendo a convergir ou divergir, para as primeiras 20 mil iterações de treinamento.

Esta implementação da SCRDet é capaz de regressar tanto uma caixa de seleção orientada (rotacionada) quanto disposta de maneira horizontal, portanto a performance da rede nas duas atividades será realizada, embora o interesse principal esteja no ângulo dos objetos.

Foi adotado uma valor para a interseção sobre a união de $IoU \geq 0,5$ para todos os testes realizados.

A SCRDet se utiliza em sua entrada de uma CNN para realizar a extração de características da imagem e, para todas as etapas do projeto optou-se por utilizar a ResNet com 101 camadas. A escolha da rede se deu em função da ResNet ser robusta e apresentar resultados excelentes e a quantidade de camadas escolhidas foi 101 e não 152 (versão de melhor acurácia) por uma questão de limitação de hardware.

Além disso, foi utilizada a técnica de transferência de aprendizado. A ResNet 101 sempre foi inicializada a partir de pesos oriundos de treinamento realizado para o banco de dados DOTA.

Todos os gráficos para o processo de treinamento apresentarão os dados brutos do erro (perda) obtidos em azul e uma curva em vermelho, que é a regressão logarítmica dos dados, a qual mostra a tendência de convergência do algoritmo.

4.3.1 Treino

A primeira etapa da aplicação do método SCRDet se deu com a utilização do banco de dados inicial (subseção 4.1.2). Uma vez que foi observada uma distribuição irregular dos objetos das classes ao longo do banco de dados, foi escolhida a técnica de valização cruzada (subseção 3.1.2.4) através da divisão dos dados em 5 dobras.

Por motivo de simplificação da apresentação dos resultados, será apresentado aqui apenas o gráfico de treinamento referente à uma das dobras tendo sido retirada para teste. Os outro gráficos, embora diferentes, tem características bem semelhantes.

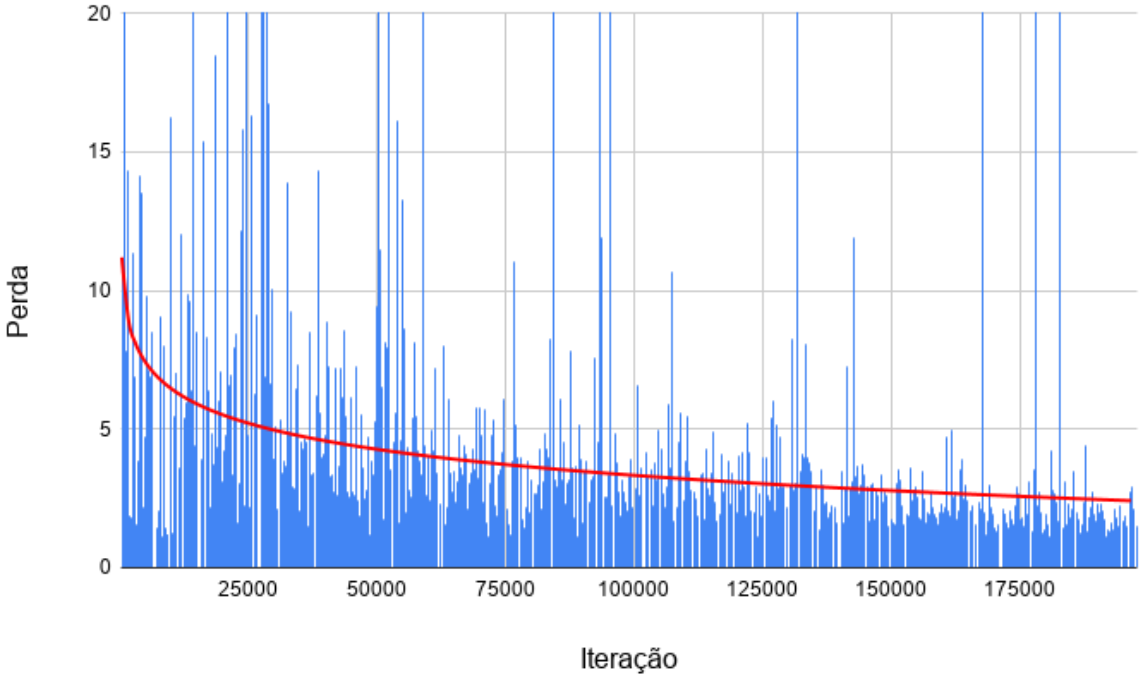
Para o treino foi utilizada uma taxa de aprendizado de LR inicialmente.

4.3.1.1 Resultados

Após o treinamento ter sido realizado utilizando todas as 5 dobras como validação, foi realizado o levantamento das métricas de avaliação do modelo (subseção 3.1.2.7).

Na TABELA 5 temos o levantamento da precisão média (AP) para as 5 dobras sendo utilizadas como teste para o caso das detecções realizadas regressando uma caixa de selação horizontal.

FIGURA 47 – TREINAMENTO: DOBRA 3



FONTE: O autor.
LEGENDA: Treinamento realizado utilizando a dobra 3 como validação.

O resultado final para média das precisões médias do modelo é **mAP** @ 0,5: $0,41058 \pm 0,063$.

TABELA 5 – AP DA DETECÇÃO HORIZONTAL

Classe	Dobra 1	Dobra 2	Dobra 3	Dobra 4	Dobra 5
Poste	0,6156	0,4637	0,4139	0,5272	0,6261
Chave Fechada	0,6463	0,7166	0,3922	0,5813	0,6083
Chave Aberta	0,2000	0,1000	0,1193	0,0159	0,1323
mAP @ 0,5	0,4873	0,4267	0,3085	0,3748	0,4556

FONTE: O autor.
LEGENDA: Valor da AP obtido para cada dobra como validação.

Na TABELA 5 temos o levantamento da precisão média (AP) para as 5 dobras sendo utilizadas como teste para o caso das detecções realizadas regressando uma caixa de selação rotacionada.

O resultado final para média das precisões médias do modelo é **mAP** @ 0,5: $0,36586 \pm 0,053$.

TABELA 6 – AP DA DETECÇÃO ORIENTADA

Classe	Dobra 1	Dobra 2	Dobra 3	Dobra 4	Dobra 5
Poste	0,5295	0,4771	0,4556	0,4370	0,5168
Chave Fechada	0,6067	0,6800	0,3795	0,5352	0,5554
Chave Aberta	0,2000	0,0167	0,0421	0,0000	0,0562
mAP @ 0,5	0,4454	0,3912	0,2924	0,3241	0,3762

FONTE: O autor.
LEGENDA: Valor da AP obtido para cada dobra como validação.

4.3.1.2 Avaliação

De acordo com os resultados, a precisão média (AP) se mostra superior para a classe de chave fechada. Na sequência, o melhor resultado fica para a classe poste e por fim, para a classe chave aberta. Observando as características do banco de dados (subseção 4.1.2), percebe-se uma relação entre a qualidade dos resultados e quantidade de exemplos para cada classe de objeto.

Sendo assim, partiu-se para a hipóte-se de que a ampliação do banco de dados seria uma boa estratégia para melhorar os resultados da média das precisões médias (mAP), o qual para este caso se mostrou baixo.

Além disso, ao longo da primeira etapa percebeu-se a lentidão em se realizar o treinamento do modelo (em torno de dois dias para cada dobra). Sendo assim, por uma questão de tempo disponível para a finalização do projeto, fica clara a necessidade de adotar outra técnica de avaliação.

4.3.2 Treino com banco de dados ampliado

A segunda etapa da aplicação do método SCRDet se deu com a utilização do banco de dados ampliado (subseção 4.1.3).

Uma vez que temos uma distribuição melhor dos objetos ao longo do banco de dados e ainda é necessário reduzir o tempo da etapa de treinamento, optou-se por dividir o conjunto de dados em duas partições.

No total, foram direcionadas 80% das imagens para o conjunto de treino e 20% para o conjunto de validação. Além disso, estas imagens foram embaralhadas antes de sua separação, com o objetivo de garantir que nenhum dos conjuntos possuisse uma proporção superior de objetos de uma determinada classe.

A seguir, na TABELA 7, vemos como estão distribuídos os dados em cada uma das partições. Foi calculada a porcentagem de objetos de cada classe para o total de objetos naquela partição (treino ou teste) afim de observar que o embaralhamento dos dados foi eficiente.

TABELA 7 – CONFIGURAÇÃO DOS CONJUNTOS

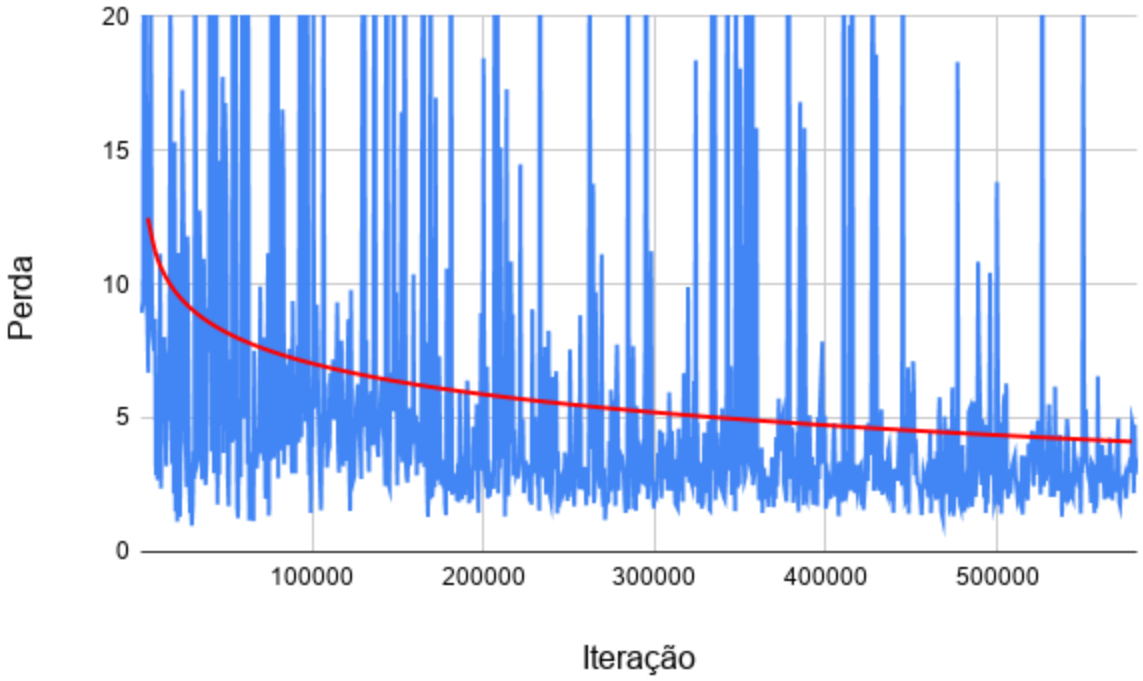
Classe	Quantidade	Treino	Treino (%)	Teste	Teste (%)
Poste	3261	2589	58,01	672	57,83
Chave Fechada	2071	1654	37,06	417	35,89
Chave Aberta	293	220	4,93	73	6,28
Total	5625	4463	100	1162	100

FONTE: O autor.
LEGENDA: Distribuição dos objetos nos conjuntos de treino e teste.

A seguir, na FIGURA 48, vemos a curva do treinamento realizado para esta etapa do projeto. É possível observar uma oscilação maior ao longo das iterações, uma vez que agora o modelo possui uma quantidade maior de dados aos quais precisa aprender a se ajustar.

Foi utilizada uma taxa de aprendizado inicial de LR

FIGURA 48 – TREINAMENTO COM BANCO DE DADOS AMPLIADO

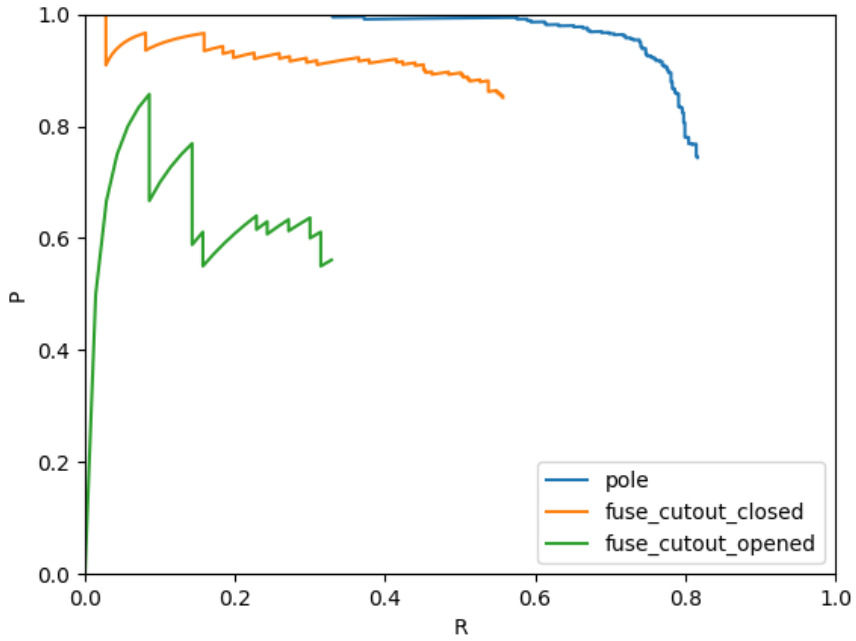


FONTE: O autor.
LEGENDA: Registro do processo de treinamento.

4.3.2.1 Resultados

A seguir, temos os gráficos da precisão (P) pela revocação (R) para a tarefa de detecção para as caixas de seleção horizontais (FIGURA 49).

FIGURA 49 – PRECISÃO POR REVOCAÇÃO - HORIZONTAL



FONTE: O Autor
LEGENDA: Gráfico da precisão pela revocação.

A seguir, temos os gráficos da precisão (P) pela revocação (R) para as tarefas de detecção para as caixas de seleção orientadas (FIGURA 50).

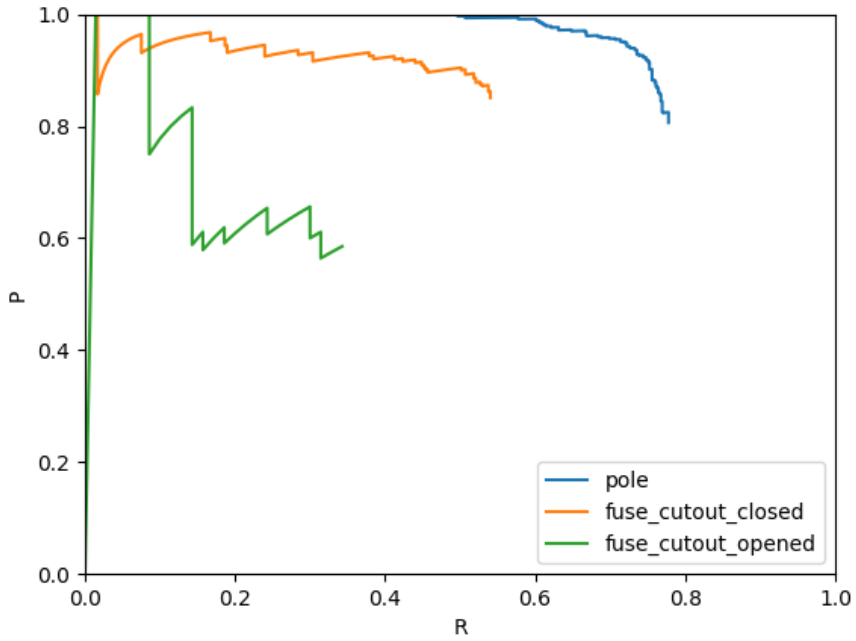
As precisões médias (APs) obtidas através do conjunto de testes para esta etapa do projeto podem ser verificadas na TABELA 8, na qual foram compilados os resultados tanto para a tarefa de detecção horizontal quando orientada.

TABELA 8 – AP RESULTANTE

Classe	Orientado	Horizontal
Poste	0,7675	0,8009
Chave Fechada	0,5074	0,5186
Chave Aberta	0,2619	0,2345
mAP @ 0,5	0,5113	0,5180

FONTE: O autor.
LEGENDA: Resultado obtido para o conjunto de teste.

FIGURA 50 – PRECISÃO POR REVOCAÇÃO - ORIENTADO



FONTE: O Autor
LEGENDA: Gráfico da precisão pela revocação.

4.3.2.2 Avaliação

Conforme era esperado, houve melhoria nos resultados após a ampliação do banco de dados. Contudo os resultados ainda se mostraram insatisfatórios para o caso das chaves fechadas.

A composição dos resultados indicou a necessidade em aumentar a variabilide dos dados desta classe e, além disso, fez com que surgisse a suspeita de que melhorando o balanceamento dos dados também seria útil para a melhoria dos resultados.

4.3.3 Treino com banco de dados ampliado e aumento de dados

A terceira etapa da aplicação do método SCRDet se deu com a utilização do banco de dados com aumento de dados e balanceamento de classes (subseção 4.1.4).

Da mesma forma que na segunda etapa, também foi realizada a separação de 80% das imagens do banco de dados para o conjunto de treino e 20% para o conjunto de validação. E, além disso, as imagens foram também foram embaralhadas antes de sua separação. A seguir, na TABELA 9 observamos a configuração destes dados.

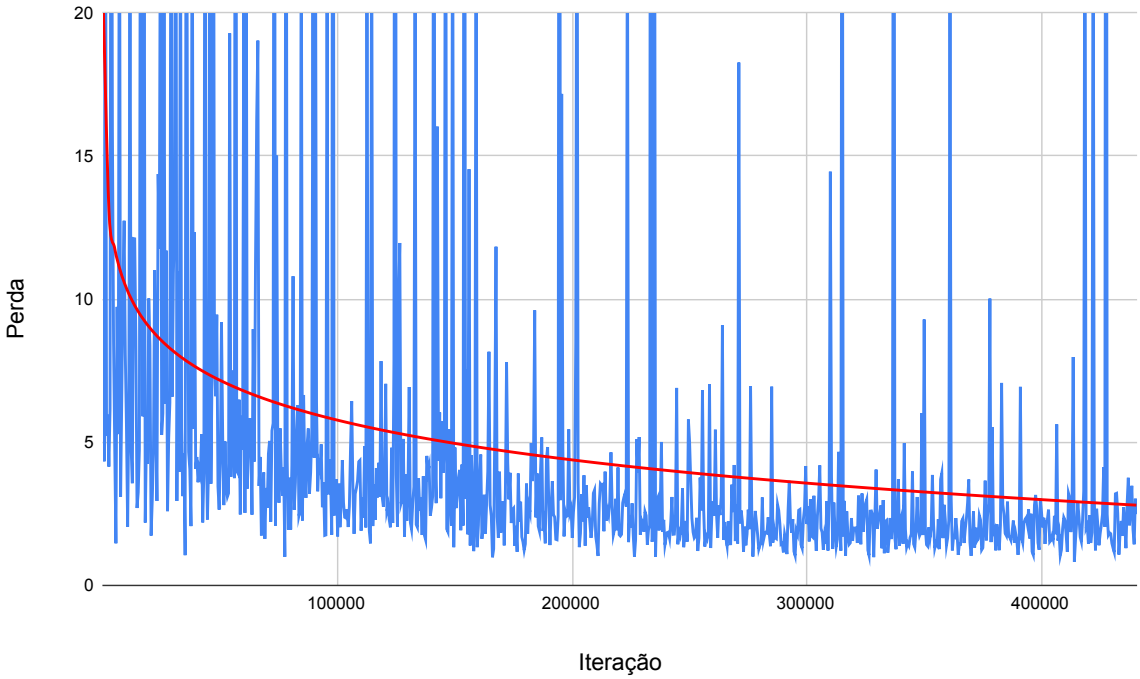
A seguir, na FIGURA 51, temos o gráfico do treinamento para esta terceira etapa de implementação da SCRDet.

TABELA 9 – COMPOSIÇÃO FINAL DOS CONJUNTOS DE TREINO E TESTE

Classe	Quantidade	Treino	Treino (%)	Teste	Teste (%)
Poste	2210	1754	44,29	456	45,51
Chave Fechada	1953	1579	39,87	374	37,33
Chave Aberta	799	627	15,83	172	17,17
Total	4962	3960	100	1002	100

FONTE: O autor.
LEGENDA: Distribuição dos objetos nos conjuntos de treino e teste.

FIGURA 51 – TREINAMENTO FINAL



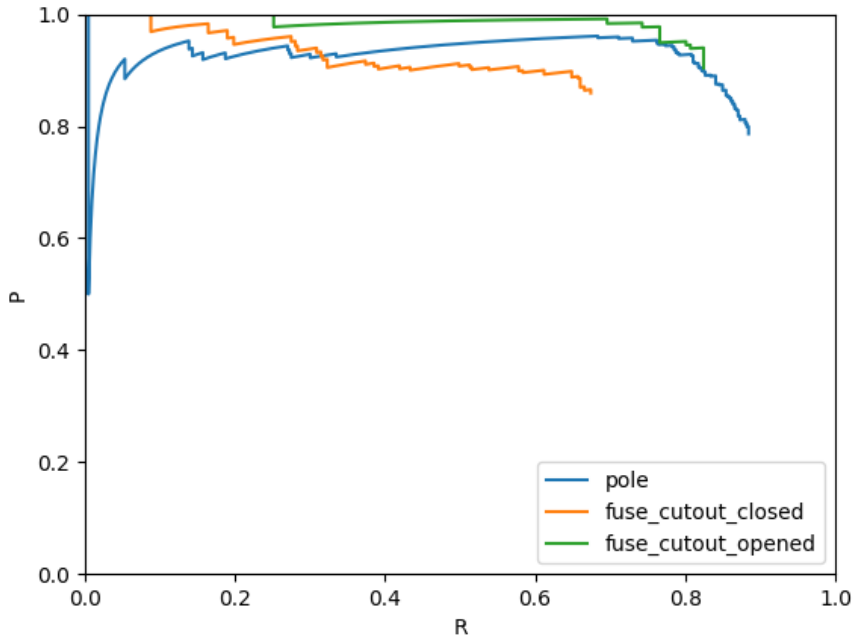
FONTE: O autor.
LEGENDA: Registro do treinamento final realizado.

Em comparação com a segunda etapa, vemos como a rede converge para um ponto em que a perda (erro/custo) é menor.

4.3.3.1 Resultados

A seguir, temos os gráficos da precisão (P) pela revocação (R) para a tarefa de detecção para as caixas de seleção horizontais (FIGURA 52).

FIGURA 52 – PRECISÃO POR REVOCAÇÃO FINAL - HORIZONTAL.



FONTE: O autor.
LEGENDA: Gráfico da precisão pela revocação.

A seguir, temos os gráficos da precisão (P) pela revocação (R) para a tarefa de detecção para as caixas de seleção horizontais (FIGURA 53).

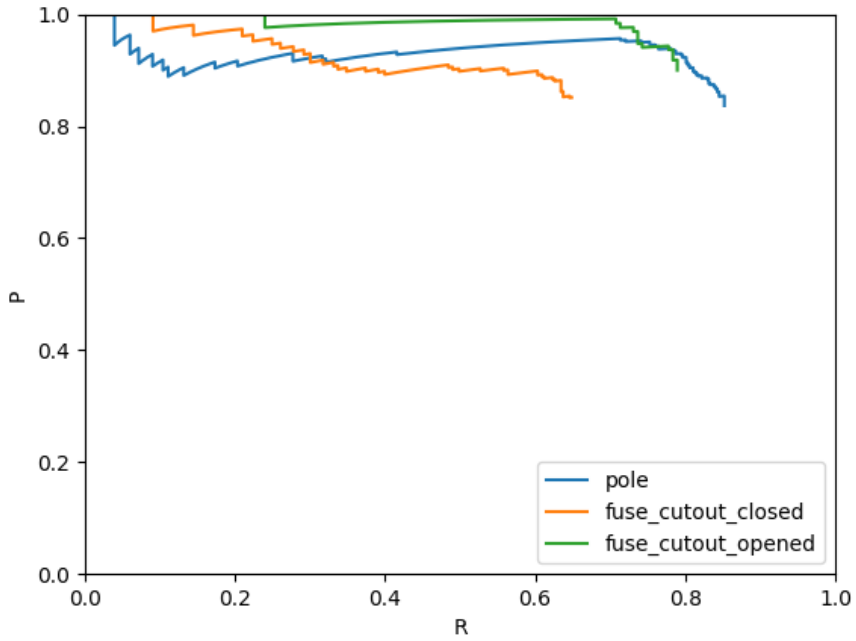
As precisões médias (APs) obtidas através do conjunto de testes para esta etapa do projeto podem ser verificadas na TABELA 10, na qual foram compilados os resultados tanto para a tarefa de detecção horizontal quanto orientada.

TABELA 10 – AP FINAL

Classe	Orientado	Horizontal
Poste	0,8122	0,8416
Chave Fechada	0,6076	0,6329
Chave Aberta	0,7818	0,8165
mAP @ 0,5	0,7338	0,7637

FONTE: O autor.
LEGENDA: Resultado final obtido para o conjunto de teste.

FIGURA 53 – PRECISÃO POR REVOCAÇÃO FINAL - ORIENTADO.



FONTE: O autor.
LEGENDA: Gráfico da precisão pela revocação.

4.3.3.2 Avaliação

Os resultados mostram como o modelo se beneficiou do aumento de dados e do balanceamento de classes. Além disso, os resultados obtidos nesta terceira etapa para o mAP ultrapassam a casa do 0,7.

Aqui também foi mantida a tendência da detecção que resulta em caixas de seleção horizontais ser ligeiramente melhor do que para o caso das caixas de seleção orientadas. De certa forma isto é esperado, uma vez que há uma complexidade maior envolvida fazer a regressão para o caso em esta acompanha a rotação do objeto.

Neste ponto optou-se por concluir a etapada de aprendizado do método SCRDet.

4.3.4 Resultados da Inferência

Nesta seção serão apresentados exemplos dos resultados da inferência realizada aplicando-se o conjunto de teste ao modelo resultante da terceira e última etapa de implementação da SCRDet.

Os resultados foram divididos em três grupos pelo autor: grupo 1 (ANEXO C), quando todos os objetos são detectados e localizados perfeitamente; grupo 2 (ANEXO D), quando no geral o resultado é bom mas a rede falhou em alguns pequenos pontos; e grupo 3 (ANEXO E), nos quais a rede falhou completamente.

Será possível observar nas imagens as caixas de seleção encontradas pela rede, bem

como a classificação dos objetos (”pole”, para os postes, ”fuse_cutout_closed”, para chaves fechadas e ”fuse_cutout_opened”, para chaves abertas) e a confiança que a rede tem de que aquela classificação está correta.

A inferência foi realizada também de maneira paralela em três conjuntos contendo 100 imagens cada. Cada conjunto possuia imagens de apenas uma mesma resolução; São elas: 1920 x 1080, 1280 x 720 e 720 x 480 pixels.

Na média, a rede demorou em torno de 220 ms para processar cada imagem, independente da resolução.

5 CONCLUSÃO E TRABALHOS FUTUROS

5.1 DISCUSSÃO

Foi realizada uma pesquisa com o intuito de averiguar o desenvolvimento das técnicas de detecção de objetos em imagens, focando na detecção de postes e chaves. Embora pouco conteúdo específico tenha sido desenvolvido nesta área, a pesquisa mostrou a evolução da utilização de métodos clássicos de aprendizado de máquina para o campo do aprendizado futuro. O levantamento teórico realizado sobre as CNNs mostrou a possibilidade da aplicação deste tipo de tecnologia para o problema endereçado neste projeto.

Uma vez que a forma dos postes, de maneira generalista, costuma ser semelhante, optou-se por utilizar o método desenvolvido para a detecção de objetos em imagens aéreas SCRDet para que fosse possível detectar suas orientações na imagem. Além disso, esta rede ainda se mostra boa para detectar objetos com dimensões distintas, como é o caso do postes e das chaves, imagens com bastante ruído no plano de fundo, como os ambientes em que estes objetos costumam se encontrar, e em configurações densas, como as chaves podem aparecer.

Para a implementação do método escolhido, não foi encontrado um banco de dados de domínio público com imagens das classes de objetos de interesse, o que culminou na necessidade da construção de um banco de dados do zero. Esta etapa se mostrou desafiadora dada a quantidade de esforço laboral envolvida, como na obtenção dos registros e na rotulação dos dados.

Ao longo da etapa de implementação, foi verificado que o processo de treinamento era computacionalmente custoso, o que levou a decisão de mudança das técnicas de avaliação.

Os resultados inciais embora não tenha se mostrado tão bons, indicavam que a ideia de utilizar a SCRDet para a tarefa era viável. Aplicando a metodologia e as técnicas levantadas na fundamentação teórica, foi possível melhorar o modelo da CNN, chegando a um ponto em que a aplicação realiza bem a tarefa de detecção de postes, com suas orientações, e das chaves fusível com seus estados. Contudo, os resultados ainda indicam que seria possível avançar na qualidade do modelo, continuando com a ampliação e diversificação do banco de dados, até que as métricas de avaliação atinjam um ponto de saturação.

Como não há um banco de dados de referência, os resultados só podem ser comparados com os obtidos para as diferentes configurações de implementação da própria rede. Mas, de qualquer forma, é possível observar, tanto através das métricas de avaliação finais, quanto dos resultados visuais obtidos, que a utilização de aprendizado profundo e processamento de imagens digitais é viável para lidar com o problema definido no projeto.

Por fim, foi avaliada a velocidade de processamento da rede. Mesmo em diferentes resoluções o resultado obtido foi muito próximo. Isto ocorre uma vez que a rede redimensiona a imagem de entrada para um tamanho máximo. Embora ela tenha se mostrado de certa forma

veloz, mais testes, conduzidos em outros tipos de hardware, possibilitariam uma compreensão maior sobre sua performance na questão de processamento.

5.2 TRABALHOS FUTUROS

A ideia inicial do desenvolvimento deste trabalho se baseia na premissa de que dado o aumento da qualidade das câmeras e a diminuição de seu custo nos últimos anos, cada vez mais as áreas urbanas estarão cobertas por vigilância por vídeo. Isto já tem acontecido em diversas metrópoles mundiais, seja por motivos de segurança púbica, controle e gerenciamento de trâfego, e assim por diante. Além disso, cada vez mais as cidades tem construído centros de processamentos destes dados, uma vez que as cidades tem se tornado mais inteligentes e conectadas. Sendo assim, a informação em tempo real contendo imagens das ruas, com os postes e chaves, já estaria disponível. Seria uma então uma questão de como os governos locais trabalharia em conjunto com as concessionárias para a disponibilização destes dados.

O cenário comentado anteriormente tem se mostrado cada vez mais possível, mas esta ainda está longe de ser a realidade da maior parte das áreas urbanas ao redor do mundo.

Sendo assim, para trabalhos futuros estão a implementação de outros métodos e a realização de um comparativo entre eles, levando em conta a qualidade da detecção e a velocidade de inferência dos dados. Na sequência, o desenvolvimento de um hardware embarcado, contendo câmera e unidade de processamento gráfico, como protótipo de um produto que pode ser utilizado pelas concessionárias, tanto em áreas urbanas quanto em áreas rurais, em estrutura fíxa ou em veículos. Na sequência, a realização de um estudo de viabilidade econômica da produção e operação destes sistemas.

Além disso, para os trabalhos futuros está a ampliação contínua do banco de dados, além da inclusão de outras classes, como de transformadores, por exemplo, que ampliam o escopo de possibilidades de aplicação do sistema desenvolvido neste projeto.

ANEXOS

ANEXO A – ANOTAÇÃO PADRÃO PASCAL VOC

```
<annotation verified="no">
  <folder>postes-chaves</folder>
  <filename>0002</filename>
  <path>/home/restani/Documents/postes-chaves/0002.png</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>3120</width>
    <height>4160</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <type>robndbox</type>
    <name>pole</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <robndbox>
      <cx>1559.5</cx>
      <cy>2109.8821</cy>
      <w>1857.9034</w>
      <h>3550.0377</h>
      <angle>0.41</angle>
    </robndbox>
  </object>
  <object>
    <type>robndbox</type>
    <name>fuse_cutout_closed</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <robndbox>
      <cx>2188.953</cx>
      <cy>1669.129</cy>
      <w>265.8938</w>
      <h>356.2082</h>
```

```
        <angle>0.99</angle>
      </robndbox>
    </object>
</annotation>
```

ANEXO B – ANOTAÇÃO PADRÃO SCRDET

```
<annotation verified="no">
  <folder>postes-chaves</folder>
  <filename>0004</filename>
  <path>/home/restani/Documents/postes-chaves/0004.png</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>4160</width>
    <height>3120</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <type>robndbox</type>
    <name>pole</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
        <x0>2576</x0><y0>2954</y0>
        <x1>2113</x1><y1>3000</y1>
        <x2>1860</x2><y2>476</y2>
        <x3>2322</x3><y3>429</y3>
    </bndbox>
  </object>
  <object>
    <type>robndbox</type>
    <name>fuse_cutout_closed</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
        <x0>2070</x0><y0>841</y0>
        <x1>2014</x1><y1>877</y1>
        <x2>1957</x2><y2>789</y2>
        <x3>2013</x3><y3>752</y3>
    </bndbox>
```

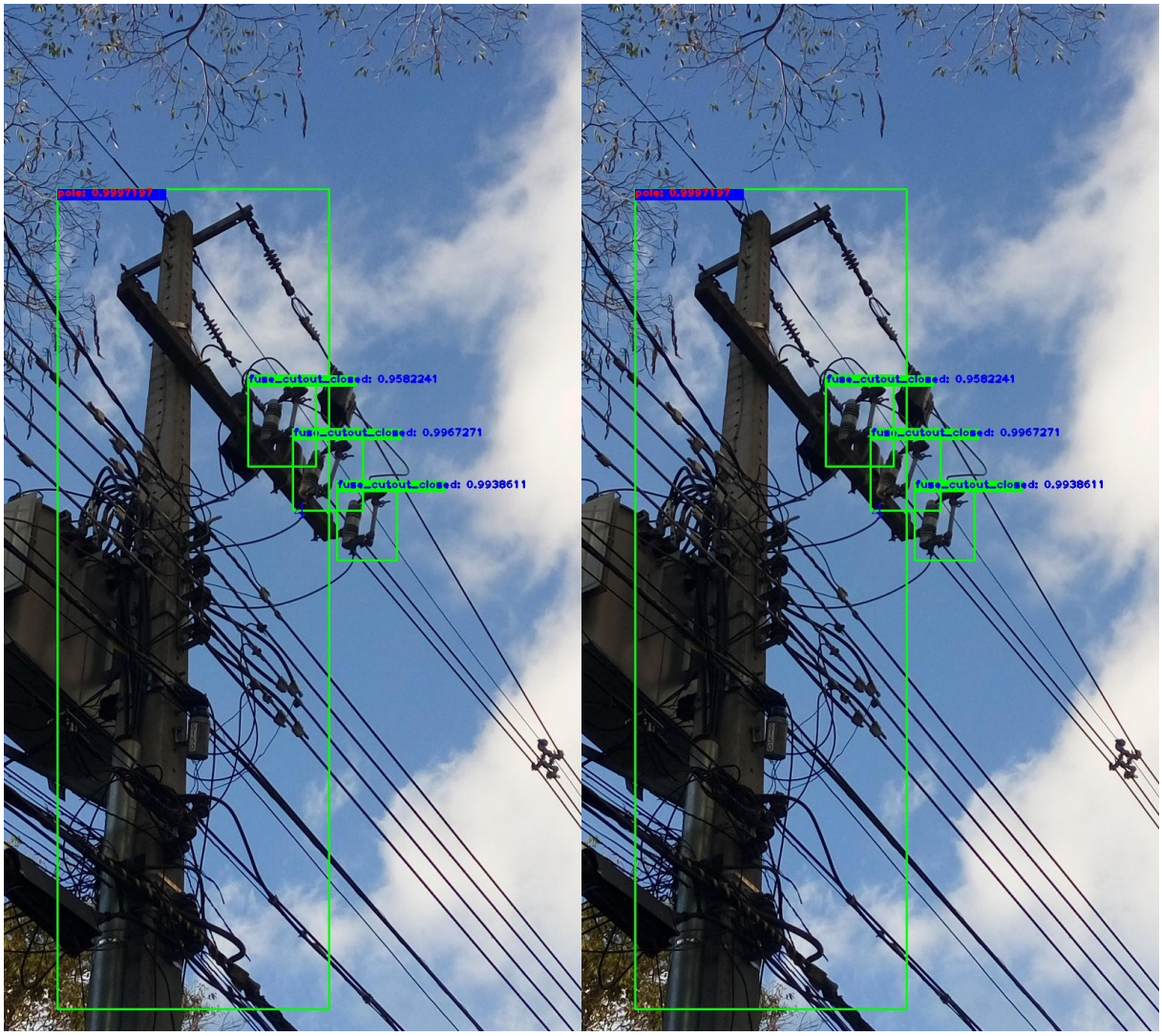
```
    </object>
</annotation>
```

ANEXO C – GRUPO 1

FIGURA 54 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 55 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 56 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 57 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção e regressão. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 58 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 59 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 60 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 61 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 62 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 63 – SCRDET: RESULTADO

Autor.

FONTE: 1
NOTA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).
LEGENDA:

FIGURA 64 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

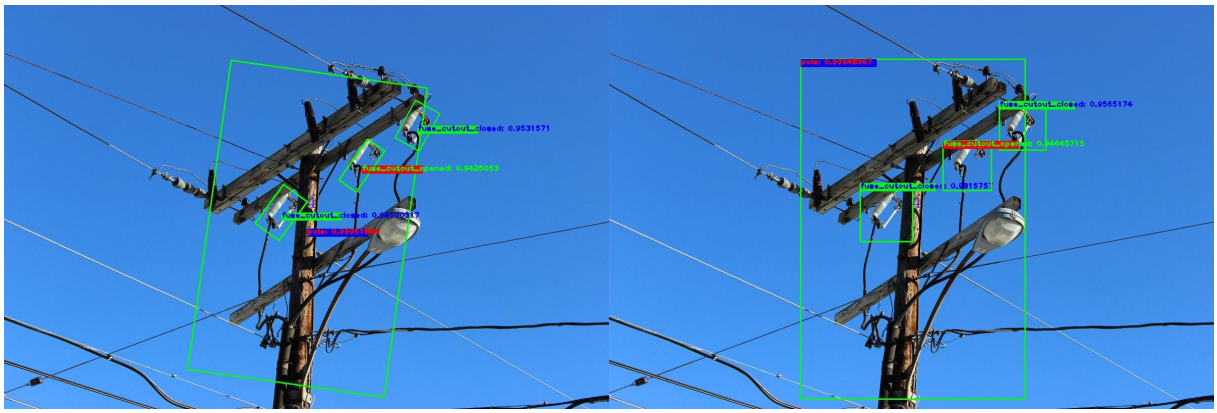
FIGURA 65 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 66 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 67 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 68 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

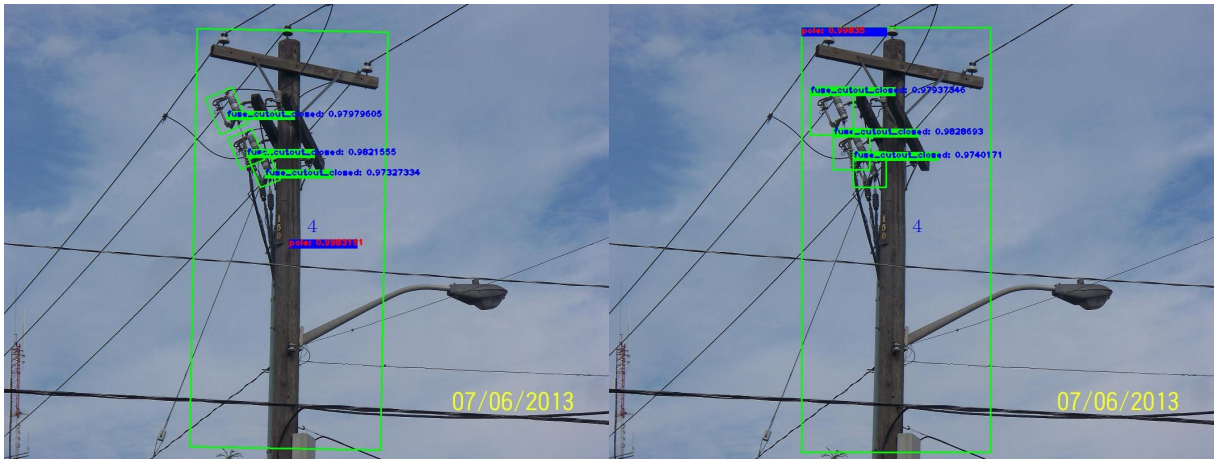
FIGURA 69 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 70 – SCRDET: RESULTADO

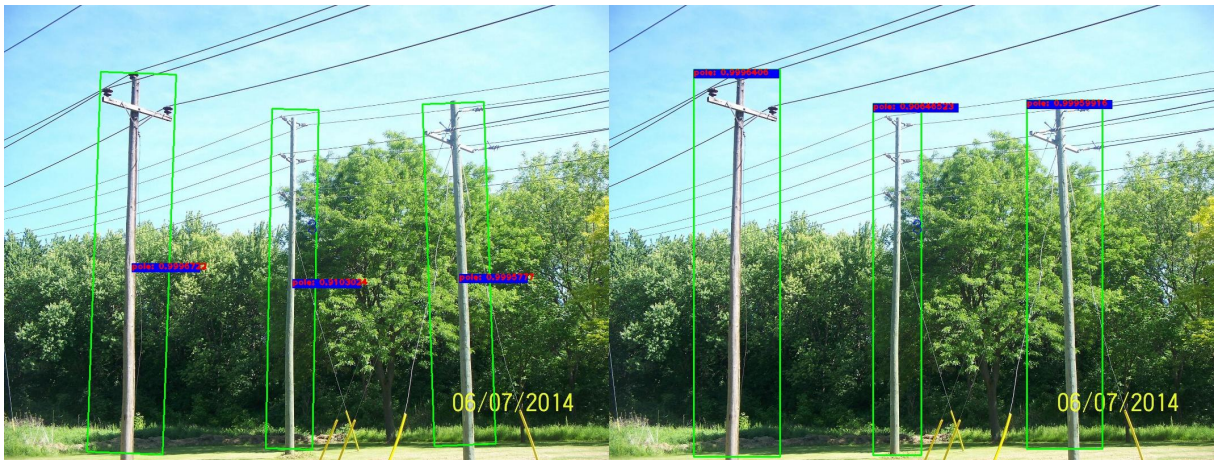
FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 71 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 72 – SCRDET: RESULTADO

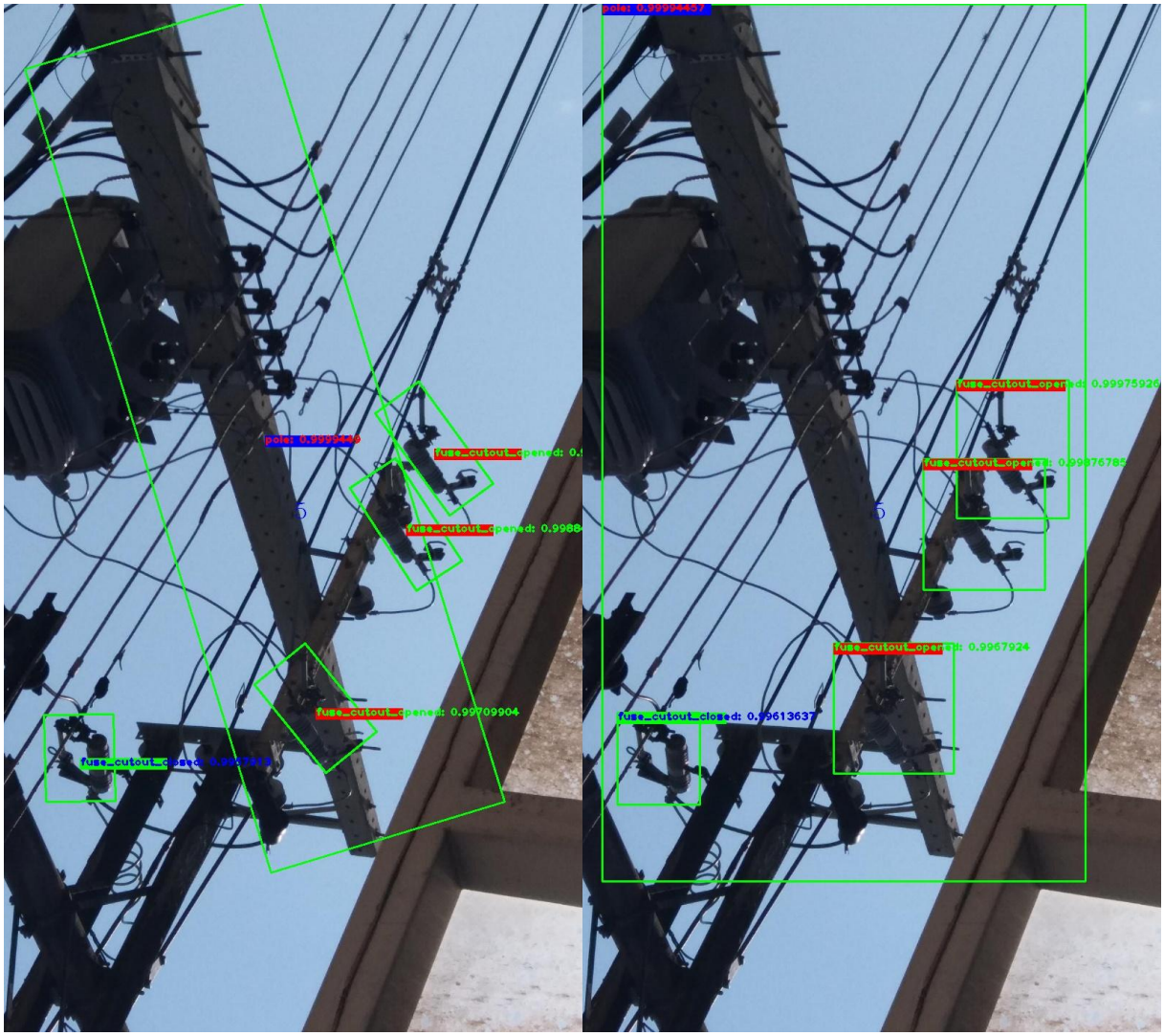


FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 73 – SCRDET: RESULTADO

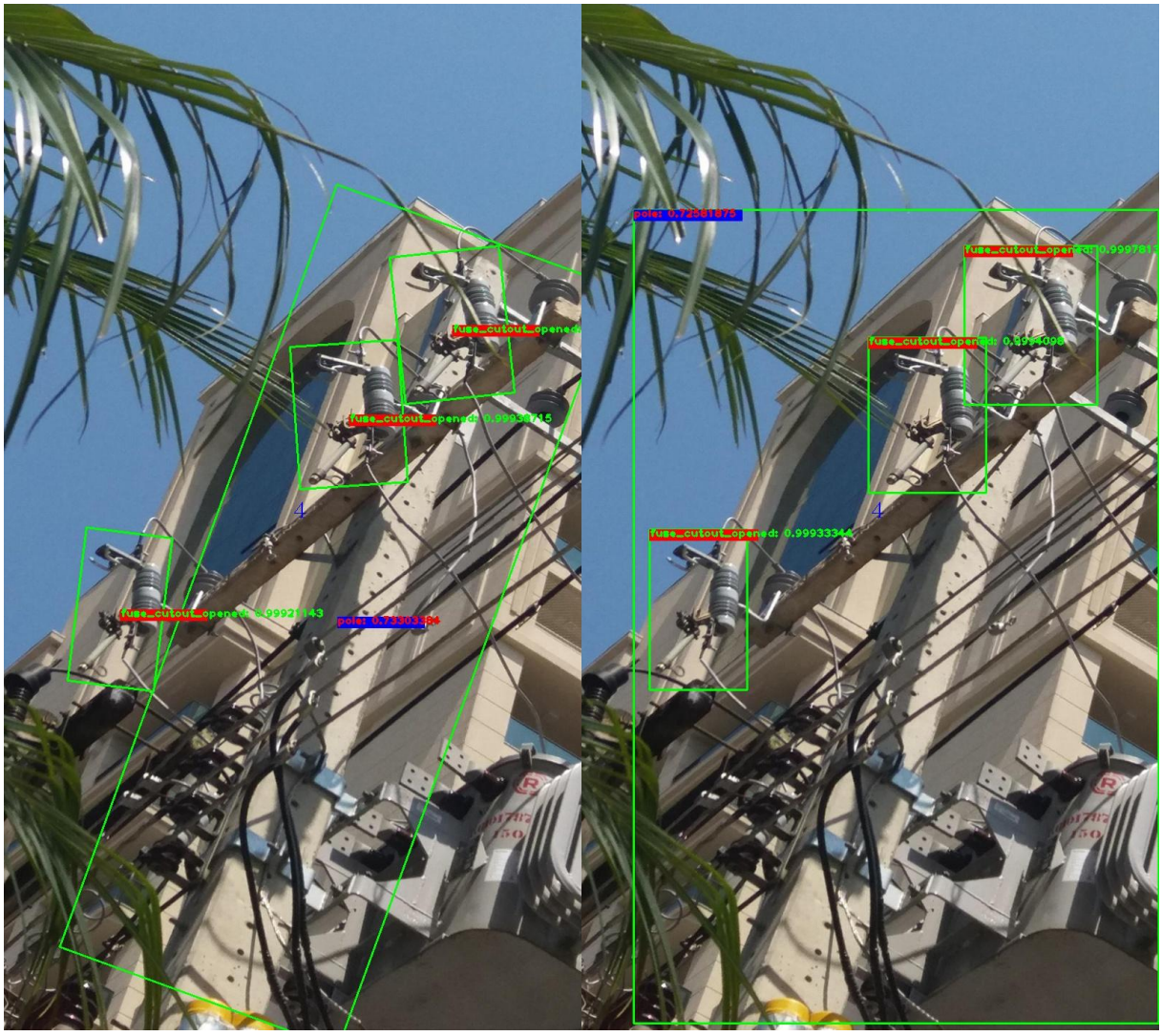
FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 74 – SCRDET: RESULTADO



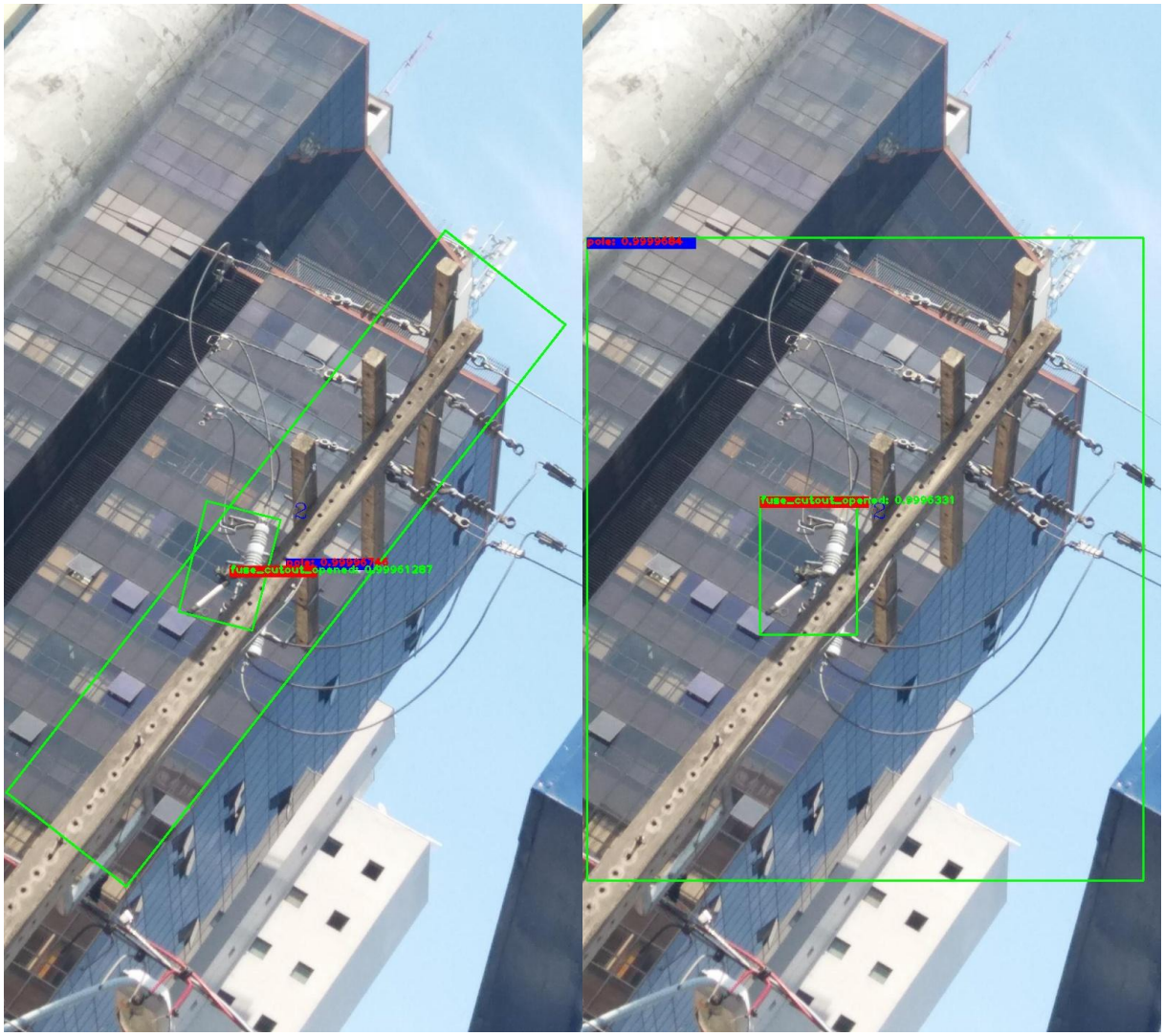
FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 75 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 76 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

ANEXO D – GRUPO 2

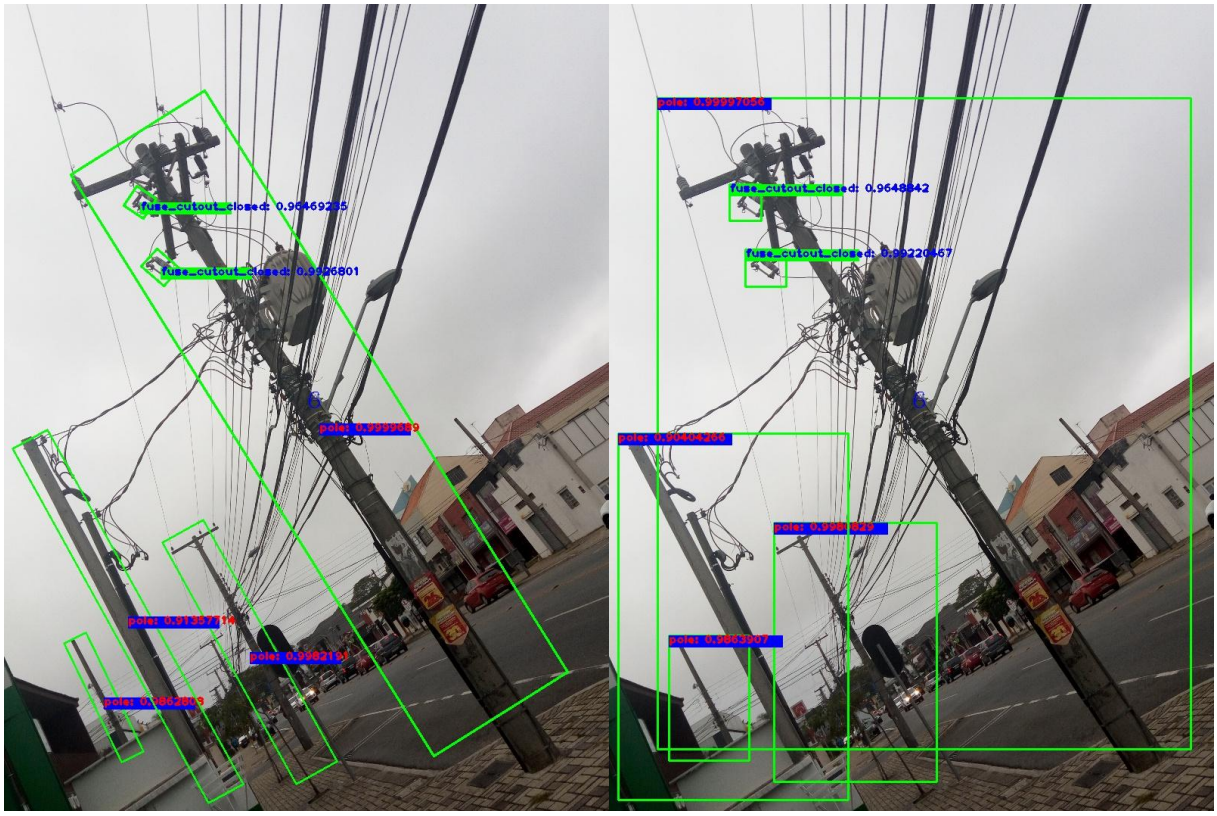
FIGURA 77 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 78 – SCRDET: RESULTADO

FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 79 – SCRDET: RESULTADO

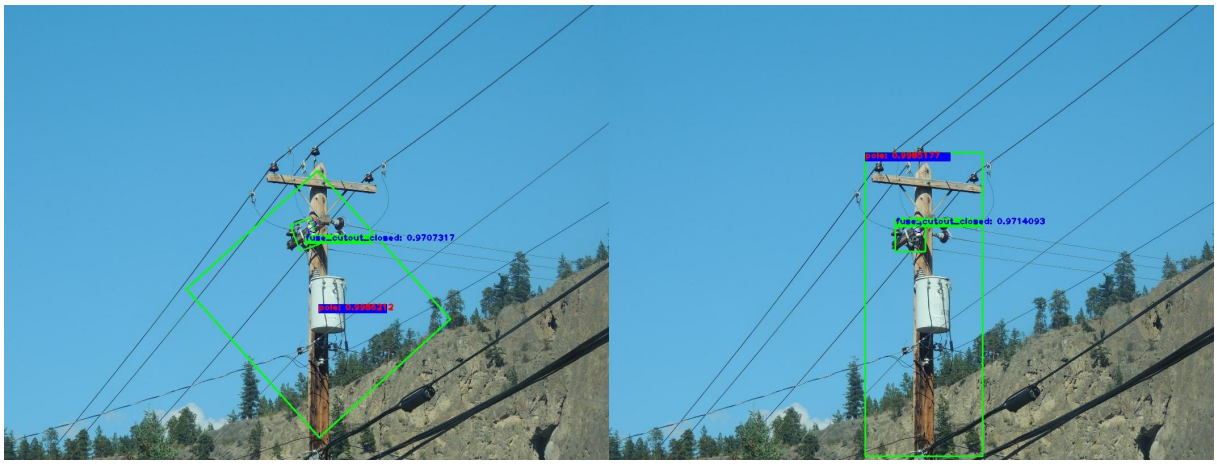


FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 80 – SCRDET: RESULTADO

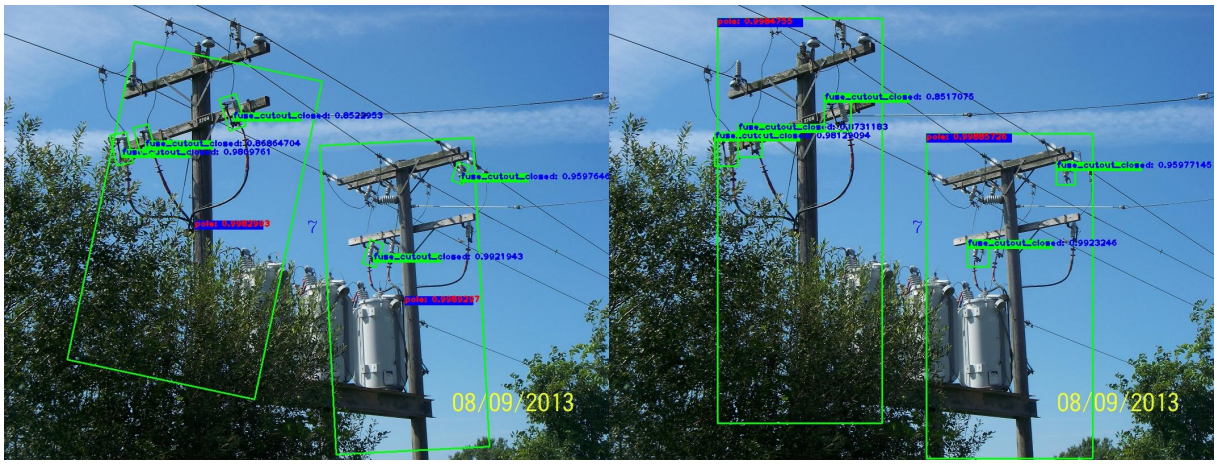
FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 81 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 82 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção e regressão. Caixas de seleção orientadas (esquerda) e horizontais (direita).

ANEXO E – GRUPO 3

FIGURA 83 – SCRDET: RESULTADO

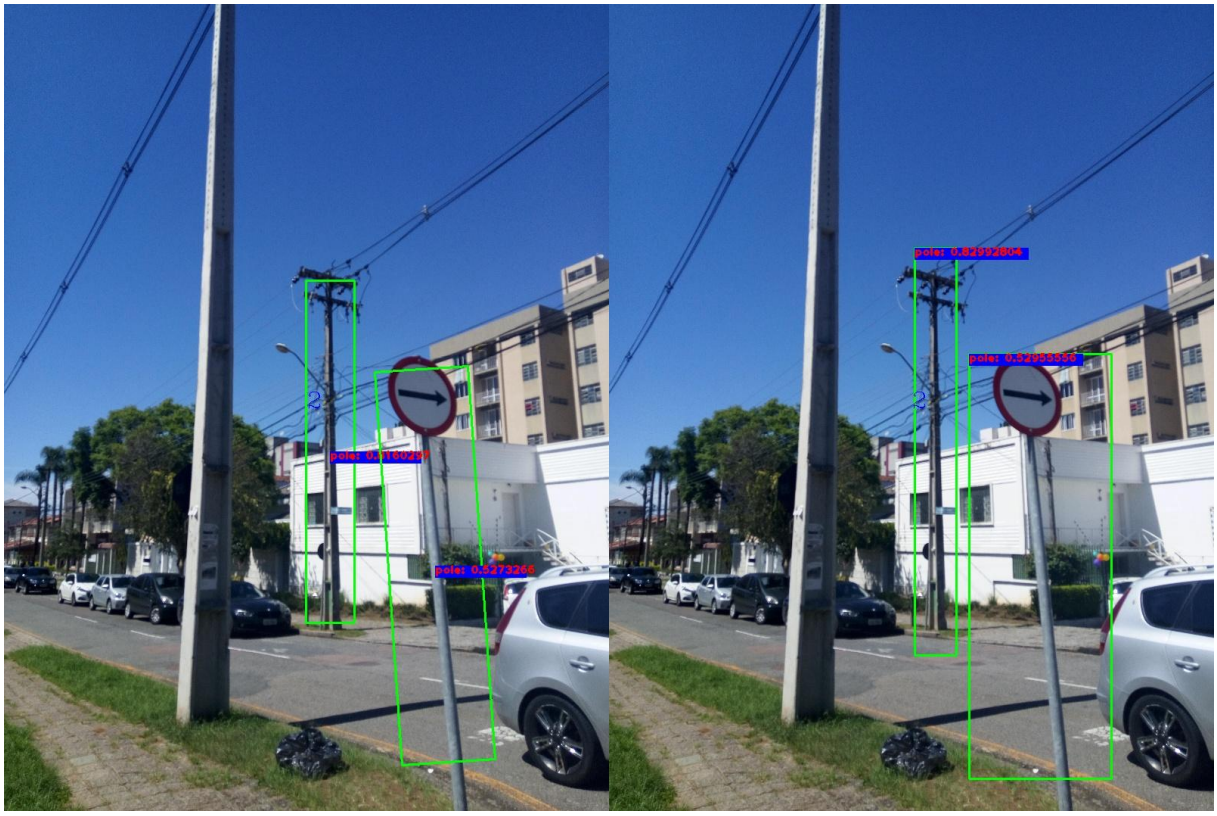
FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

figura SCRDet: Resultado 1 fig/g3/7 O Autor. 1 Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 84 – SCRDET: RESULTADO

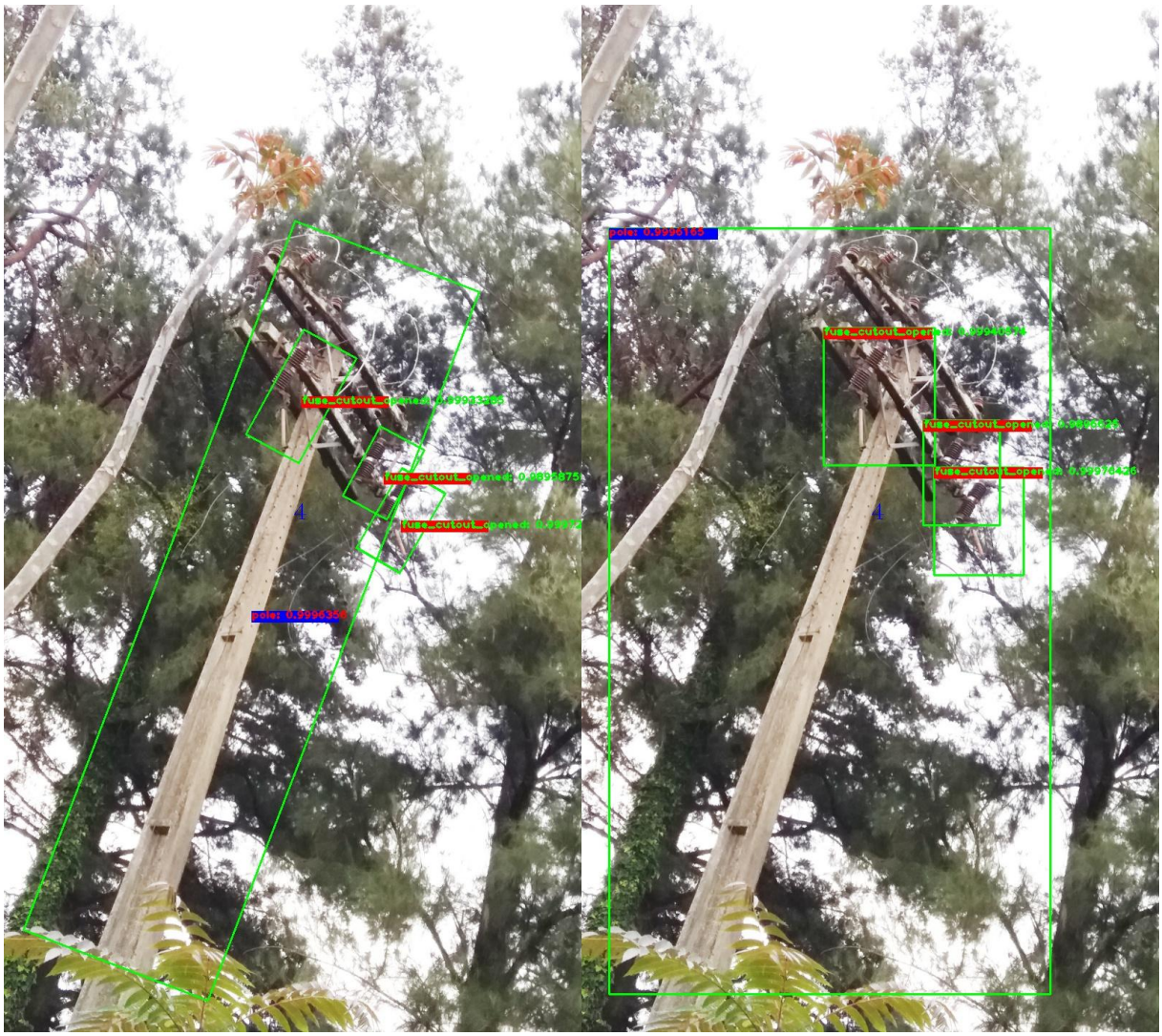
FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 85 – SCRDET: RESULTADO



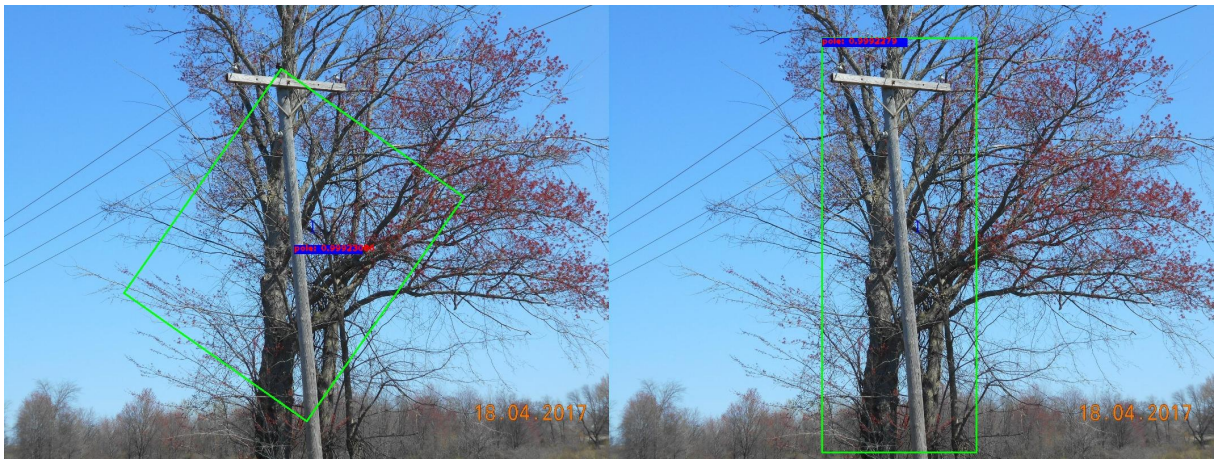
FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 86 – SCRDET: RESULTADO



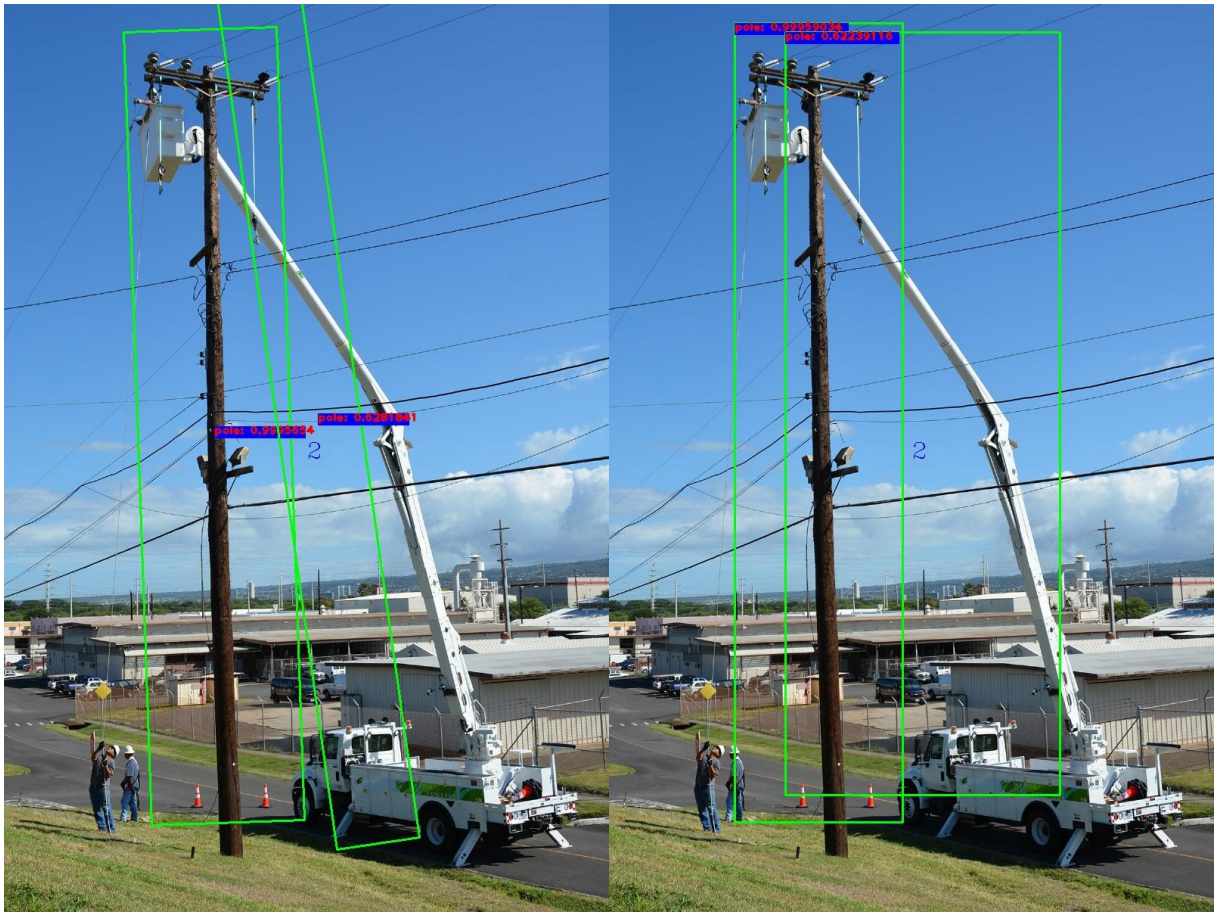
FONTE: O Autor.
LEGENDA: Detecção e regressão. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 87 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

FIGURA 88 – SCRDET: RESULTADO



FONTE: O Autor.
LEGENDA: Detecção. Caixas de seleção orientadas (esquerda) e horizontais (direita).

REFERÊNCIAS

BLAGA, Bianca Cerasela Zelia; NEDEVSCHI, Sergiu. A method for automatic pole detection from urban video scenes using stereo vision. In: PROCEEDINGS - 2018 IEEE 14th International Conference on Intelligent Computer Communication and Processing, ICCP 2018. 2018. p. 293–300. Citado 1 vez na página 19.

BROWN, Richard. **Electric Power Distribution Reliability, Second Edition**. CRC Press, set. 2008. v. 20081087, p. 504. (Power Engineering (Willis)). Citado 3 vezes nas páginas 14, 15.

CETIN, B.; BIKDASH, M.; MCINERNEY, M. Automated electric utility pole detection from aerial images. In: CONFERENCE Proceedings - IEEE SOUTHEASTCON. 2009. p. 44–49. Citado 1 vez na página 19.

CHEN, Binghuang; MIAO, Xiren. Distribution Line Pole Detection and Counting Based on YOLO Using UAV Inspection Line Video. **Journal of Electrical Engineering and Technology**, jul. 2019. Citado 1 vez na página 20.

CHENG, Gong et al. Multi-class geospatial object detection and geographic image classification based on collection of part detectors. **ISPRS Journal of Photogrammetry and Remote Sensing**, 2014. Citado 1 vez na página 60.

CHENG, Wengang; SONG, Zhengzheng. Power pole detection based on graph cut. In: PROCEEDINGS - 1st International Congress on Image and Signal Processing, CISP 2008. 2008. v. 3, p. 720–724. Citado 1 vez na página 18.

DENG, Jia et al. ImageNet: A large-scale hierarchical image database. In: p. 248–255. Citado 2 vezes nas páginas 54, 56.

FUKUSHIMA, Kunihiko; MIYAKE, Sei. Neocognitron: A Self-Organizing Neural Network Model for a Mechanism of Visual Pattern Recognition. In: 1982. Citado 1 vez na página 46.

GÉRON, Aurélien. **Hands–On Machine Learning with Scikit–Learn and TensorFlow**. O'Reilly, 2017. p. 566. Citado 1 vezes nas páginas 24, 27.

GIRSHICK, Ross. Fast R-CNN. In: PROCEEDINGS of the IEEE International Conference on Computer Vision. 2015. Citado 1 vez na página 59.

GIRSHICK, Ross et al. Rich feature hierarchies for accurate object detection and semantic segmentation. In: PROCEEDINGS of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2014. Citado 1 vez na página 59.

GOLIGHTLY, Ian; JONES, Dewi. Corner detection and matching for visual tracking during power line inspection. **Image and Vision Computing**, v. 21, n. 9, p. 827–840, 2003. Citado 1 vez na página 18.

_____. Visual control of an unmanned aerial vehicle for power line inspection. In: 2005 International Conference on Advanced Robotics, ICAR '05, Proceedings. 2005. v. 2005, p. 288–295. Citado 1 vez na página 18.

GONZALEZ, Rafael; WOODS, Richard. **Processamento de Imagens Digitais**. Edgard Blücher, 2000. p. 509. Citado 1 vez na página 47.

GOODFELLOW, Ian et al. Generative Adversarial Networks. **Advances in neural information processing systems**, p. 2672–2680, 2014. Citado 1 vez na página 57.

GUIMARÃES, Ana. **Confiabilidade de Sistemas de Distribuição: Calibração de Dados e Mecanismos para Avaliação de Desempenho**. 2006. f. 193. Tese (Doutorado) – Universidade Federal de Itajubá. Citado 1 vez na página 15.

HAHNIOSER, Richard H.R. et al. Digital selection and analogue amplification coexist in a cortex- inspired silicon circuit. **Nature**, 2000. Citado 1 vez na página 51.

HAYKIN, Simon. **Neural Networks and Learning Machines**. 2008. v. 3, p. 906. Citado 2 vezes nas páginas 22, 25.

HE, Kaiming et al. Deep residual learning for image recognition. In: PROCEEDINGS of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2016. Citado 1 vezes nas páginas 56, 57.

HU, Jie; SHEN, Li; SUN, Gang. Squeeze-and-Excitation Networks. In: PROCEEDINGS of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2018. Citado 1 vez na página 57.

HUANG, Jonathan et al. Speed/accuracy trade-offs for modern convolutional object detectors. In: PROCEEDINGS - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017. 2017. 2017-Janua, p. 3296–3305. Citado 1 vez na página 57.

HUBEL, D. H.; WIESEL, T. N. Receptive fields of single neurones in the cat's striate cortex. **The Journal of Physiology**, 1959. Citado 1 vez na página 46.

JONES, D. I. Aerial inspection of overhead power lines using video: Estimation of image blurring due to vehicle and camera motion. **IEE Proceedings: Vision, Image and Signal Processing**, v. 147, n. 2, p. 157–166, 2000. Citado 1 vez na página 18.

JONES, Dewi I. et al. A laboratory test-bed for an automated power line inspection system. **Control Engineering Practice**, v. 13, n. 7, p. 835–851, 2005. Citado 1 vez na página 18.

JONES, DI; WHITWORTH, CC; DULLER, AWG. Image processing methods for the visual location of power line poles. In: PROC 7th Irish Machine vision and Image Processing conference (IMVIP2003). 2003. p. 177–184. Citado 1 vez na página 18.

KARATZAS, Dimosthenis et al. ICDAR 2015 competition on Robust Reading. In: PROCEEDINGS of the International Conference on Document Analysis and Recognition, ICDAR. 2015. Citado 1 vez na página 60.

KHAN, Salman et al. **A Guide to Convolutional Neural Networks for Computer Vision**. 1. ed.: Morgan & Claypool, 2018. p. 207. Citado 5 vezes nas páginas 32, 40, 47–51.

KRIZHEVSKY, Alex; SUTSKEVER, Ilya; HINTON, Geoffrey E. ImageNet classification with deep convolutional neural networks. In: ADVANCES in Neural Information Processing Systems. 2012. v. 2, p. 1097–1105. Citado 1 vez na página 54.

LECUN, Yann; BENGIO, Yoshua; HINTON, Geoffrey. **Deep learning**. v. 521. 2015. p. 436–444. Citado 1 vez na página 20.

LECUN, Yann et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, 1998. Citado 1 vezes nas páginas 54, 55.

LECUN, Y. et al. Backpropagation Applied to Handwritten Zip Code Recognition. **Neural Computation**, 1989. Citado 1 vez na página 47.

LI, Zhengrong et al. Knowledge-based power line detection for UAV surveillance and inspection systems. In: 2008 23rd International Conference Image and Vision Computing New Zealand, IVCNZ. 2008. Citado 1 vez na página 19.

LIN, Tsung Yi; GOYAL, Priya et al. Focal Loss for Dense Object Detection. In: PROCEEDINGS of the IEEE International Conference on Computer Vision. 2017. 2017-Octob, p. 2999–3007. Citado 3 vezes nas páginas 63, 64.

LIN, Tsung Yi; MAIRE, Michael et al. Microsoft COCO: Common objects in context. In: PART 5. LECTURE Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2014. 8693 LNCS, p. 740–755. Citado 2 vezes nas páginas 56, 63.

LIU, Wei et al. SSD: Single shot multibox detector. In: LECTURE Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2016. Citado 1 vez na página 62.

MARTINEZ, Carol et al. Towards autonomous detection and tracking of electric towers for aerial power line inspection. In: 2014 International Conference on Unmanned Aircraft Systems, ICUAS 2014 - Conference Proceedings. 2014. p. 284–295. Citado 1 vez na página 19.

MCCULLOCH, Warren S; PITTS, Walter. A logical calculus nervous activity. **Bulletin of Mathematical Biology**, v. 52, n. 50, p. 99–115, dez. 1990. Citado 1 vez na página 22.

NEAPOLITAN, Richard E.; NEAPOLITAN, Richard E. **Neural Networks and Deep Learning**. Determination Press, 2018. p. 389–411. Citado 1 vez na página 56.

NIELSEN, Michael A. **Neural Networks and Deep Learning**. Determination Press, 2015. Citado 4 vezes nas páginas 28, 29, 31–33, 35.

NORDENG, Ian E. et al. DEBC detection with deep learning. In: LECTURE Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2017. 10269 LNCS, p. 248–259. Citado 1 vez na página 19.

OZDEMIR, Sinan. **Principles of Data Science**. Packt Publishing, 2017. p. 388. Citado 1 vez na página 37.

REDMON, Joseph et al. You only look once: Unified, real-time object detection. In: PROCEEDINGS of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2016. Citado 1 vezes nas páginas 62, 63.

REN, Shaoqing et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 2017. Citado 1 vezes nas páginas 59, 60.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, p. 386–408, 1958. Citado 1 vez na página 22.

RUMELHART, David E.; HINTON, Geoffrey E.; WILLIAMS, Ronald J. Learning representations by back-propagating errors. **Nature**, v. 323, n. 6088, p. 533–536, 1986. Citado 1 vez na página 32.

SAMPEDRO, Carlos et al. A supervised approach to electric tower detection and classification for power line inspection. In: PROCEEDINGS of the International Joint Conference on Neural Networks. 2014. p. 1970–1977. Citado 1 vez na página 19.

SARKAR, Dipanjan; BALI, Raghav Bali; GHOSH, Tamoghna. **Hands-On Transfer Learning with Python Implement Advanced Deep Learning and Neural Network Models Using TensorFlow and Keras**. Packt Publishing, 2018. p. 438. Citado 1 vez na página 39.

SHARMA, Hrishikesh et al. Image Analysis-Based Automatic Utility Pole Detection for Remote Surveillance. In: 2015 International Conference on Digital Image Computing: Techniques and Applications, DICTA 2015. 2015. Citado 1 vez na página 19.

SIMONYAN, Karen; ZISSERMAN, Andrew. VGG-16. **arXiv preprint**, 2014. Citado 1 vez na página 57.

STEIGER, Olivier; LUCAS, Erwan; MARET, Yannick. Automatic detection of transmission towers. In: December. PROCEEDINGS of IEEE Sensors. 2014. 2014-Decem, p. 1034–1037. Citado 1 vez na página 19.

SZEGEDY, Christian et al. Going deeper with convolutions. In: PROCEEDINGS of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2015. Citado 1 vez na página 57.

TILAWAT, Jittichat; AUEPHANWIRIYAKUL, Sansanee; THEERA-UMPON, Nipon. Automatic detection of electricity pylons in aerial video sequences. In: ICEIE 2010 - 2010 International Conference on Electronics and Information Engineering, Proceedings. 2010. v. 1. Citado 1 vez na página 19.

TOMBARI, Federico et al. Automatic detection of pole-like structures in 3D urban environments. In: IEEE International Conference on Intelligent Robots and Systems. 2014. p. 4922–4929. Citado 1 vez na página 19.

XIA, Gui Song et al. DOTA: A Large-Scale Dataset for Object Detection in Aerial Images. In: PROCEEDINGS of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2018. p. 3974–3983. Citado 1 vezes nas páginas 60, 65, 66.

XIE, Saining et al. Aggregated residual transformations for deep neural networks. In: PROCEEDINGS - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017. 2017. Citado 1 vez na página 57.

YANG, Xue et al. SCRDet: Towards More Robust Detection for Small, Cluttered and Rotated Objects. In: THE IEEE International Conference on Computer Vision (ICCV). 2019. p. 8232–8241. Citado 0 vezes nas páginas 61, 62, 65.

ZHANG, Weixing et al. Using deep learning to identify utility poles with crossarms and estimate their locations from google street view images. **Sensors (Switzerland)**, v. 18, n. 8, 2018. Citado 1 vez na página 20.

ZHAO, Zhong Qiu et al. **Object Detection With Deep Learning: A Review**. 2019. Citado 1 vez na página 54.

ZOU, Zhengxia et al. Object Detection in 20 Years: A Survey. **arXiv preprint arXiv:1905.05055**, maio 2019. Citado 1 vezes nas páginas 20, 58.