UNIVERSIDADE FEDERAL DO ABC
CENTRO DE ENGENHARIA, MODELAGEM E CIÊNCIAS SOCIAIS APLICADAS
ENGENHARIA DE INFORMAÇÃO

EDUARDO HENRIQUE DOS SANTOS MARQUES

APRENDIZADO DE MÁQUINA E PROCESSAMENTO DE IMAGENS PARA PLANEJAMENTO DE PODAS COMO MEDIDA PREVENTIVA DE FALHAS NO SISTEMA DE DISTRIBUIÇÃO DE ENERGIA ELÉTRICA

Santo André - SP, Abril de 2021

EDUARDO HENRIQUE DOS SANTOS MARQUES

Aprendizado de máquina e processamento de imagens para planejamento de podas como medida preventiva de falhas no sistema de distribuição de energia elétrica

Trabalho apresentado como requisito para a conclusão do curso de Engenharia de Informação da Universidade Federal do ABC.

Universidade Federal do ABC – UFABC

Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas

Engenharia da Informação

Orientador: Prof. Dr. Luneque Del Rio de Souza e Silva Junior

Santo André - SP

Abril de 2021

Agradecimentos

Agradeço aos familiares, pelo apoio, contribuindo para elaboração deste trabalho.

Agradeço ao Prof. Dr. Luneque Del Rio de Souza e Silva Junior, orientador deste trabalho, pelo acompanhamento, sugestões, instruções e esclarecimentos

Agradecimentos à empresa Daimon Engeharia e Sistemas e membros pela disponibilização de dados e também por sugestões e esclarecimentos em questões do projeto.

Resumo

O trabalho teve como objetivo investigar uma metodologia automática baseada em técnicas de aprendizado de máquina em conjunto com processamento de imagens permitindo definir um plano para realização de podas. Para isso foi feito um estudo sobre técnicas de aprendizado de máquina, em especial aprendizagem profunda, aplicadas a problemas de visão computacional (classificação, detecção de objetos e segmentação). Além disso, um estudo sobre a diversidade de espécies de vegetação e sobre normas e especificações para manejo e cuidados da arborização e das interferências causadas no sistema de distribuição de energia. Finalmente, foi realizada a implementação de um sistema que facilite nas decisões do processo de poda da vegetação.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

API	*Application Programamming Interface*
BCE	*Binary Cross Entropy*
CAFFE	*Convolutional Architecture for Fast Feature Embedding*
CRF	*Conditional Random Field*
CNN	*Convolutional Neural Network*
DEF	*Destritores Elipticos de Fourier*
DL	*Deep Learning*
DwC	*Darwin Core Standard*
FPN	*Feature Pyramid Network*
GBIF	*Global Biodiversity Information Facility*
GE	*Google Earth*
GIS	*Geographic Information System*
GSV	*Google Street View*
GPU	*Graphics Processing Unit*
GPS	*Global Positioning System*
IoU	*Intersection Over Union*
JI	*Jaccard Index*
Lidar	*Light Detection And Ranging*
LOB	*Line of Bearing*
MLP	*Multilayer Perceptron*
NMS	*Non Maximum Supression*
PDI	Processamento Digital de Imagens
PFH	*Feature Pyramid Hierarchy*

RDA	Rede de distribuição aérea convencional
RDP	Rede de distribuição aérea compacta
RDI	Rede de distribuiçãoaérea isolada
RDS	Rede de distribuição subterrânea
ReLU	*Rectified Linear Unit*
ANN	*Artificial Neural Network*
ROC	*Receiver Operating Characteristic*
UPC	*Utility Pole with Crossarms*
VANT	Veículo Aéreo Não Tripulado
VC	Visão Computacional

Sumário

1 Introdução

A energia elétrica é um recurso essencial na atualidade, e, por este motivo, no Brasil, como em outros países temos agentes que se responsabilizam pelo fornecimento de energia. No entanto, podemos observar que no Brasil centros de produção de energia são muitas vezes distantes das regiões de consumo. Deste modo, o fornecimento pode ser dificultado, existindo a necessidade de contratação de serviços de distribuidoras de energia [5].

Um problema básico enfrentado pelas distribuidoras são as interrupções do fornecimento de energia por desligamentos ou outras interferências provocadas por diferentes agentes, sendo que, embora existam métodos consolidados que permitem monitorar, através de estimativas, a ocorrência e localização destas falhas. Ainda são poucos os métodos que permitem reduzir as causas de falhas e desligamentos na rede elétrica, principalmente pelo fato de muitas destas causas resultarem de fatores imprevisíveis. Dentre estes fatores alguns exemplos são chuvas fortes, ventos, raios, batidas de veículos em postes, contato de folhas e galhos de árvores e atos de vandalismo. Deve-se observar que já existe um conjunto de etapas de trabalho que permitem reduzir estes fatores, como trabalhos de manutenção, construção de suporte e estruturas para proteção das redes. Sendo que para os problemas causados pela vegetação, a principal é a poda de árvores. No entanto, atualmente esse processo ainda é feito de forma manual, e este pode ser um trabalho difícil, afinal, existe um conjunto amplo de características e regras, neste caso particularmente da vegetação, que precisam ser consideradas para conservação da biodiversidade, garantindo a sustentabilidade deste processo. Portanto, é interessante que este trabalho possa ser realizado de forma automatizada [2][6].

Conjuntamente a estas necessidades, as áreas de estudo em sensoriamento remoto, visão computacional (VC) e processamento de imagens, têm passado por diversas mudanças nos últimos anos. A respeito da área de sensoriamento remoto, vêm sendo encontradas novas formas para exploração de diferentes ambientes, como veículos aéreos não tripulados (VANTs). Por outro lado, as áreas de visão computacional e processamento de imagem tem passado por grandes mudanças e quebras de paradigmas nos últimos 5 a 10 anos. Isso porque, devido a maiores capacidades de processamento de dados adquiridas recentemente, com o uso de paralelismo em mais núcleos de processamento das *"Graphics Processing Unit"* (GPU), ou obtidos através de clusterização, a tecnologia de redes neurais voltou a ser aplicada a problemas de aprendizado de máquina e tem apresentado resultados interessantes, especialmente quando consideradas os métodos de *Deep Learning* (DL) [7][8].

1.1 Objetivos

O objetivo deste trabalho é investigar uma metodologia automática baseada em técnicas de aprendizado de máquina em conjunto com processamento de imagens permitindo definir um plano para realização de podas. Com isso será possível avaliar problemas como invasão de vegetação, através da proximidade das árvores com os elementos de infraestrutura de distribuição de energia, e características como as distâncias entre vegetação e estes elementos, verificando se estão nos intervalos permitidos pelas normas do setor elétrico.

1.2 Organização do Trabalho

O trabalho se divide nos seguintes capítulos:

Capitulo 2: Neste capítulo são apresentadas referências bibliográficas relacionadas com às áreas de interesse deste trabalho. Portanto, referências envolvendo as áreas de reconhecimento e detecção de elementos da infraestrutura de distribuição de energia, tanto a partir de algoritmos de visão computacional tradicionais, como estudos sobre a possibilidade de desenvolvimentos através de técnicas de *Deep Learning*. Também são considerados alguns trabalhos envolvendo classificação de espécies de plantas.

Capitulo 3: Neste capítulo são apresentados elementos teóricos que fundamentam o desenvolvimento do trabalho, partindo de conceitos na área de aprendizado de máquina até conceitos sobre processamento de imagens com técnicas de *Deep Learning*. Finalmente são apresentados alguns conceitos elementares sobre o sistema elétrico e o planejamento de podas.

Capitulo 4: Neste capítulo é apresentada a metodologia definida para se alcançar os objetivos do trabalho. Portanto, são considerados procedimentos e parâmetros específicos que serão utilizados nas tarefas de detecção de objetos, segmentação e classificação de espécies. Finalmente, é apresentado ainda um conjunto de experimentos para ajuste de parâmetros dos métodos citados.

Capitulo 5: Neste capítulo são apresentados os resultados para as tarefas e métodos definidos. Portanto, um conjunto de imagens e modelo de dados resultantes da aplicação dos métodos.

Capitulo 6: Neste capítulo são apresentadas conclusões sobre os procedimentos utilizados e resultados correspondentes. Também são apresentadas sugestões para futuros trabalhos.

2 Trabalhos Relacionados

Existem diversos trabalhos que lidam com o problema de processamento de imagem aplicado na diversidade de elementos presentes no ambiente urbano, como reconhecimento de edifícios, postes (de diferentes tipos), vegetação, pessoas, etc. Além disso, também existem características que diferenciam as formas como a aquisição e processamento destas imagens pode ser realizado. A respeito das diferenças presentes no processo de aquisição, estão associadas principalmente com os instrumentos usados na aquisição das imagens, como é observado em características como resolução espacial, espectral, radiométrica e temporal, em imagens obtidas por câmeras em aeronaves ou VANTs, imagens obtidas por satélites, e imagens obtidas através do mapeamento de regiões com scanners lasers *Light Detection And Ranging* (LiDAR), embora também possa haver influência do ambiente adicionando ruído ou mudanças na iluminação.

Considerando estas características, em sensoriamento remoto, a maioria das abordagens de estudo dos ambientes de interesse são baseadas no uso de imagens aéreas de alta resolução, ou seja, imagens obtidas principalmente por satélites. No entanto, recentemente, com o uso de VANTs, e além disso, o surgimento de sistemas de observação terrestre abertos, como o *Google Earth* (GE), vêm sendo propostas análises baseando-se em imagens de baixa resolução. Outra novidade é o uso de imagens terrestres de visão panorâmica, também popularizadas pela Google através da ferramenta *Google Street View* (GSV).

No trabalho [2], é realizada a identificação de postes de distribuição de energia e árvores em suas proximidade, para posteriormente verificar a periculosidade das árvores, considerando comparações entre alturas de postes e árvores. Para isso, inicialmente são utilizadas imagens aéreas obtidas com o GE para identificação de postes. A obtenção é feita através da API (Application Programming Interface) do GE. Nas requisições são passadas as coordenadas geográficas dos elementos de interesse, para os quais serão obtidas as imagens. As coordenadas foram fornecidas por uma concessionária de distribuição de energia. Isso porque, conforme apresentado no trabalho, a maioria das concessionárias possui informações de georreferenciamento deste tipo de elemento. Tendo obtido as imagens com postes, são aplicadas técnicas de processamento digital de imagens (PDI), particularmente operadores morfológicos para realizar a identificação dos postes nestas imagens. Com a identificação dos postes, é possível definir uma região de interesse na qual é feita a identificação de árvores através da aplicação de filtros (como forma de pré-processamento) e operadores morfológicos. Em um segundo momento, são utilizadas imagens de vista panorâmica obtidas com a API do GSV. Neste caso, não é feita a identificação dos postes, ao invés disso são utilizadas as coordenadas dos postes obtidas com mais precisão através da identificação nas imagens aéreas. Tendo a localização dos postes neste caso também é

possível definir uma região de interesse, esta será usada tanto para para obter relações de altura de postes, como também para identificação das árvores nas imagens frontais. Finalmente, é feito a identificação das árvores das imagens frontais, mais uma vez através de operadores morfológicos, e em seguida são utilizadas algumas hipóteses para estimar a altura das árvores. O desempenho dos métodos estudados é apresentado na tabela 1.

Tabela 1 – Métricas de Desempenho para Identificação obtidas em [2]

	Árvores em Imagens Aéreas	Postes 1 e 2 em Imagens Aéreas	Árvores em Imagens Frontais
Acurácia	88.8%	100.0%	100.0%
Sensibilidade	100.0%	100.0%	100.0%
Especificidade	9.1%	100.0%	100.0%

Observa-se que o método possui sensibilidade de identificação de 100%, das árvores em imagens aéreas, no entanto, a presença de falsos positivos que levou à diminuição de acurácia. Para o processo de identificação de postes nas duas etapas diferentes apresenta sensibilidade e acurácia de 100%, embora em alguns casos tenha apresentado o que o autor denomina de "ruídos", elementos não pertencente às classes de interesse. Os ruídos presentes podem afetar de algum modo as características analisadas, afinal, para um dos principais objetivos do trabalho, a estimativa da altura das árvores, a precisão obtida é de 74%. Foi observado dentre as dificuldades encontradas, a grande quantidade de variáveis que podem se encontrar em imagens de ferramentas como GE e GSV, dado que, possivelmente, não existe uma padronização para a captura das imagens.

No trabalho [3], é realizado o estudo de técnicas para reconhecimento de postes da rede de distribuição elétrica em imagens obtidas pelo GSV. Para isso, na obtenção das imagens foi utilizada a API do GSV. Em seguida, dentre os métodos utilizados para realizar o reconhecimento, numa etapa inicial de pré-processamento foi utilizado um método de segmentação baseado na técnica “Mean Shift” (MS) em conjunto com técnicas de estimação de densidade de probabilidade através de kernels formando clusters (regiões) de alta densidade de probabilidade que são aplicadas em conjuntos de pontos (pixels), através dessas operações os conjuntos de pixels são segmentados. O procedimento de segmentação permitiu a rotulação de segmentos resultantes nas classes pertencente ou não a postes. Após a segmentação, foi realizada a extração de características para realizar a classificação das imagens (segmentos) obtidos. Foi analisado o desempenho de classificação para classificadores treinados para reconhecer objetos através de características de cor, textura e forma, isso porque é considerado que tais características permitem a diferenciação entre postes. Para a característica de cor a extração foi baseada na moda, para textura foi baseada no uso da matriz de co-ocorrência normalizada, enquanto que para forma foi utilizado o método de Descritores Elípticos de Fourier (DEF). Na etapa seguinte foi realizado o treinamento de uma Rede Neural Artificial (ANN, do inglês, *Artificial Neural*

Network) do tipo *Multilayer Perceptron* (MLP) e, como algoritmo de otimização dos parâmetros da rede, uma variação do algoritmo de descida de gradiente chamado algoritmo Levenberg-Marquardt. Para fazer o treinamento, o conjunto de imagens foi previamente dividido pelo processo de validação cruzada, deixando parte das imagens do conjunto para realização de testes. Após o treinamento da ANN, esta foi usada no processo de classificação a partir dos métodos de extração de características considerados. Assim, o desempenho de classificação para cada método é apresentado na Tabela 2.

Tabela 2 – Métricas de Desempenho para classificação com métodos de extração de características obtidas em [3]

	Moda Cor (A)	Matriz de Co-ocorência de Textura (B)	(A) + (B)	DEF	Todas
Acurácia	0,8168	0,8586	0,8717	0,6257	0,7827
Precisão	0,7665	0,8278	0,8622	0,6043	0,7621
Sensibilidade	0,9110	0,9058	0,8848	0,7277	0,8220
Especificidade	0,7225	0,8115	0,8586	0,5236	0,7435

Foi observado que o método que utiliza características de forma, extraídas através de DEF apresentou os piores resultados. Uma explicação dada para o desempenho ruim neste caso foi a grande variedade de formas de postes presentes. Entre as outras características, o desempenho apresentado foi razoavelmente próximo (entre cor e textura) com uma pequena vantagem para o método baseado em textura. Também foi avaliado o desempenho de métodos combinados. Finalmente, considerando estes resultados, Kardec propõe um protótipo de classificador para reconhecer postes a partir das características de cor e textura, devido ao melhor desempenho utilizando estas características.

Em [7], é feita uma revisão de técnicas de visão computacional aplicadas na inspeção de transmissão de energia. Neste sentido, consideram-se como principais questões envolvidas o reconhecimento e inspeção de componentes elétricos, detecção de regiões com congelamento de elementos da rede (cenário mais comum em países de clima frio), trechos com invasão de vegetação e, finalmente, monitoramento em situações de fatalidades ou acidentes. É apresentado ainda, as principais formas de navegação autônoma, através de VANTs, que possibilitam a realização da tarefa de inspeção baseando-se na: localização por GPS, detecção de postes e detecção de linhas. Estas tarefas e os métodos aplicados apresentam alguma similaridade com um dos objetivos do trabalho, a identificação de elementos do sistema de distribuição, como as linhas e postes.

Em [7], são sugereridas as seguintes abordagens, utilizando DL, para os principais grupos de tarefas citadas:

a) Para tarefas de reconhecimento e inspeção de componentes elétricos, são sugeridos métodos de detecção de objetos por meio de redes convolucionais (CNN, do inglês, Convolutional Neural Networks), como Faster RCNN, SSD, YOLO, R-FCN, junto de redes de DL para classificação de imagem como ResNet, Inception-v4, Inception-ResNet, entre outros.

b) Para inspeção das linhas, e até mesmo de componentes, pode ser interessante utilizar segmentação para obter imagens segmentadas ou para a remoção de fundo, se necessária. Deste modo em alguns cenários, podem ser usadas abordagens de segmentação semântica, e métodos mais tradicionais de remoção de fundo, como supressão e redes neurais de pulso acoplado.

c) No monitoramento de falhas e fatalidades podem ser usados os métodos já citados, inclusive pela combinação de alguns deles, como no caso de árvores em contato com linhas, neste caso poderia ser usada uma combinação de métodos de detecção de árvores e detecção das linhas.

d) Para monitoramento de trechos com invasão de vegetação, entre outras tarefas que necessitam da detecção das linhas de transmissão, são sugeridos métodos de detecção de borda e contorno baseados em CNNs profundas, que poderão estar associadas com técnicas tradicionais como transformada de Hough.

Em [7] também são apresentadas dificuldades que podem ser encontradas para se aplicar DL neste contexto, em especial deve ser considerada a ausência de *datasets* públicos suficientemente grandes, visto que técnicas DL geralmente requerem grande volume de dados para treinamento. Sendo que, o desenvolvimento de um *dataset* próprio pode ser uma tarefa bastante árdua, como é exemplificado pelos próprios autores, que despendem de 750 horas de trabalho para o desenvolvimento de um *dataset* de forma manual. Fora o fato de a ausência de *datasets* também levar à ausência de métricas para comparar futuros resultados obtidos. Outros problemas existentes são a presença de classes desbalanceadas, devido a possibilidade de situações reais com baixa frequência ou até mesmo inexistentes no *dataset*. Dificuldade de detecção de componentes e falhas pequenas, assim como dificuldade de detecção das linhas para certas imagens, especificamente onde a separação das linhas e o fundo é difícil.

Finalmente, em [7], também é observado que a maior parte dos estudos já realizados na área referem-se a problemas específicos, utilizando principalmente de técnicas clássicas de processamento de imagem (em alguns casos também combinadas a métodos de aprendizagem de máquina), que embora possam apresentar bons resultados, em geral são muito restritos e, portanto, com pouca capacidade de generalização. Além disso, exigem grande esforço de projeto. Como alternativa, é proposto o uso de DL para o processamento de imagens. Isso porque métodos que usam DL em especial CNNS e ConvNets vêm mostrando que bons desempenhos quando aplicados em sistemas de reconhecimento de imagens, ao

mesmo tempo que não apresentam as limitações das técnicas convencionais, ou seja, grande esforço de projeto com baixa capacidade de generalização.

Em [1] inicialmente são apresentadas algumas das limitações no uso de imagens aéreas para detecção de postes em termos de resolução espacial, visto que o tamanho de postes nas imagens obtidas desta forma é relativamente pequeno. Outras limitações consideradas são que a detecção é dificultada em cenários de grande complexidade, situações com variações de iluminação, ou ainda pela cobertura de vegetação, entre outros fatores. No trabalho também são apresentadas desvantagens de métodos de mapeamento móvel como LiDAR, que embora apresente bons desempenhos, possui grande custo para obtenção dos dados, e complexidades na realização do processamento, devido ao grande número de pontos que utiliza.

Neste contexto, é proposta a detecção de postes com braços transversais (UPCs, do inglês *Utility Poles with Crossarms*) ao longo de estradas, através de imagens de vista panorâmica obtidas pelo GSV. Para isso são utilizadas técnicas DL para detecção de objetos e algoritmos que utilizam ângulo azimutal (LOB, do inglês *Line of Bearing*) como medida para estimar as localizações de UPCs detectados.

A metodologia utilizada [1], consiste, numa primeira etapa, da obtenção, pelo "United States Census Bureau", de um *dataset* de informações geográficas de estradas. Em seguida são realizados tratamentos de dados e, com auxílio da ferramenta ArcGIS, ferramenta do tipo *Geographic Information System* (GIS) especializada para análises de geolocalização e mapeamento, são obtidos 9290 pontos geográficos, com intervalos de 10m, ao longo de estradas selecionadas. Foi feita uma transformação das coordenadas utilizadas no ArcGIS, para as coordenadas usadas no GSV, dada certa incompatibilidade entre os sistemas. Finalmente, foram mapeados UPCs de referência (*ground-truth*) na região de estudo através de imagens aéreas com resolução de 7.5cm, sendo feita em seguida uma confirmação destes UPCs com imagens do GSV. Assim é obtido um conjunto de 1039 postes, presentes num entorno de 20m de pontos das estradas selecionadas. A partir destes postes foram obtidas 3500 imagens (4 imagens, para cada ponto, em ângulos de visão diferentes, com incrementos de 90^{o}), que através da rotulação (manual) de UPCs identificados, foram usadas como referência (ground-truth). As 3500 imagens foram divididas em 2500 para treino, 500 para validação e 500 para teste.

A respeito do algoritmo proposto, baseado em DL, para detecção de UPCs, pode ser dividido em 2 etapas principais:

Primeiramente, foi utilizada a rede RetinaNet para a detecção dos UPCs nas imagens obtidas do GSV. A escolha foi justificada pela estrutura mais simples e desempenho mais rápido (em alguns casos, também uma melhor acurácia), da RetinaNet em comparação com outras redes do estado da arte como a Faster R-CNN e a Mask R-CNN.

A segunda etapa foi calcular a coordenada de ângulo azimutal de cada ponto para detectar os UPCs baseando-se nos ângulos usados dentre os parâmetros das imagens do GSV. O cálculo de ângulo azimutal horizontal é feito comparando a média das coordenadas horizontais das bordas dos quadros delimitadores, resultantes do processo de detecção, com a posição central da imagem. Utilizando o mesmo procedimento para cálculo do ângulo azimutal vertical. Assim, de maneira simplificada, uma das soluções para determinação da posição dos UPCs é a intersecção entre os ângulos azimutais horizontal e vertical.

Para analisar o desempenho, e determinar um conjunto de parâmetros ótimos, no uso de DL na estimação das localizações de UPCs em imagens GSV, foram feitos experimentos em um servidor dedicado (no artigo são apresentadas as especificações de desempenho do servidor), utilizando as arquiteturas descritas abaixo.

Redes RetinaNet de 50 camadas, 101 camadas e 152 camadas, aplicadas no conjunto de imagens já rotuladas. Para todas as arquiteturas, já treinadas, foram usados como parâmetros, passo de 2500, tamanho de lote (batch size) igual a 1 e número de épocas igual a 200. Por outro lado, na etapa de treino, o tamanho do passo foi definido baseando-se nos tamanhos de lote e do *dataset* de treino. Durante o treino, foram utilizadas rotações na direção horizontal de forma aleatória para aumentação de dados. Em seguida foi realizada a validação avaliando a acurácia das 3 arquiteturas em cada época.

Como resultados, foi verificado que os algoritmos alcançaram, durante a validação, convergência (alcançando e permanecendo em seu máximo) do valor de precisão média em torno da 25ª época. Por outro lado, na avaliação da acurácia, definida na etapa de testes, foram obtidos como resultados, matrizes de confusão, conforme apresentado na Figura 1 para cada arquitetura (ou seja, RetinaNets de 50, 101 e 152 camadas), considerando valores de threshold do parâmetro *Intersection Over Union* (IoU) (também chamado índice de Jaccard), de 0.3, 0.4 e 0.5.

No trabalho foi verificado que, dentre as 3 escolhas, a arquitetura RetinaNet-101 teve melhor acurácia global, com valores de 78%, 72% e 50%, para os tresholds 0.3, 0.4 e 0.5 respetivamente. Além disso, as precisões foram de 0.95, 0.91 e 0.73, enquanto os recalls foram de 0.81, 0.77 e 0.62.

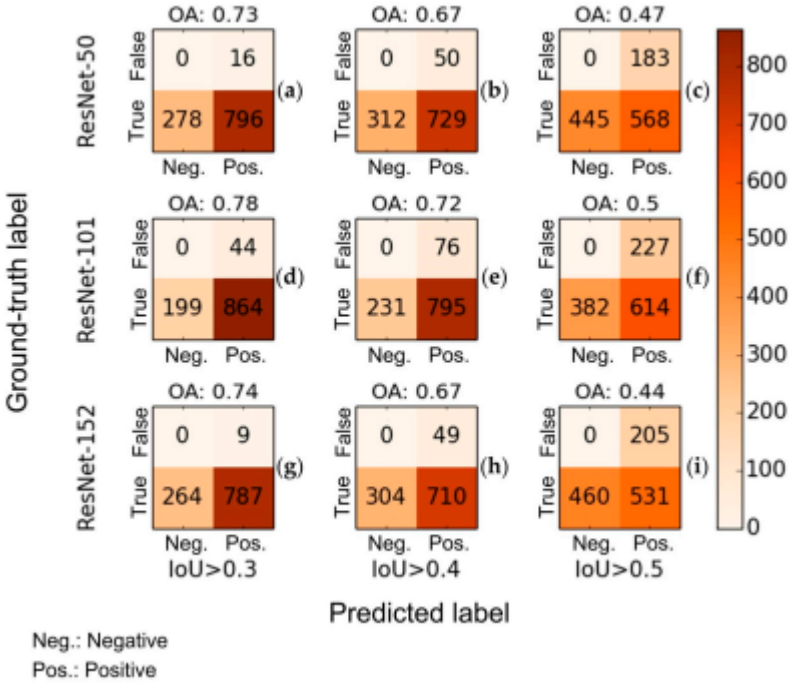
Em [9], inicialmente são apresentados diversos benefícios relacionados com a presença de árvores em determinados ambientes. Assim, tendo sido verificada a ausência de bases de dados atualizadas ou completas, é desenvolvido um sistema para construir automaticamente uma base de dados de árvores classificada e informações de interesse.

O sistema apresentado é baseado particularmente em técnicas de DL aplicadas nos problemas de detecção de objetos e classificação refinada de objetos.

O processo de detecção de árvores é dado por:

Primeiramente é realizado o *download* (em um repositório de imagens, como o

Figura 1 – Matrizes de confusão obtidas em [1], na detecção de UPCs em imagens GSV, para as arquiteturas RetinaNets de 50, 101 e 152 camadas, considerando valores de threshold do parâmetro "intersecção sobre união" de 0.3, 0.4 e 0.5.



Fonte: Referência [1]

GE ou GSV) das imagens aéreas e de vista panorâmica, correspondentes às coordenadas geográficas de pontos de uma região de interesse. Em segundo lugar, foi utilizado um esforço colaborativo (pela plataforma *Amazon Mechanical Turk*) para adicionar os rótulos ao conjunto de imagens. Em seguida foi realizada a detecção de árvores, pela diferenciação entre árvores e outras "cenas" presentes nas imagens. Para a detecção são usadas redes CNN do tipo Faster-RCNN, sendo utilizadas através do framework de redes neurais CAFFE (do inglês, Convolutional Architecture for Fast Feature Embedding). Deve-se observar que para utilizar o conjunto de imagens como entradas para as CNNS, foi realizada uma etapa de tratamento de dados, com a centralização e normalização dos valores nas imagens. Outra observação em relação a este processo é a definição de uma função que permite fazer uma transformação do domínio da localização dos pixels em uma imagem para o domínio de coordenadas geográficas. Deste modo, a rede Faster RCNN é aplicada nas coordenadas geográficas de regiões de interesse. O trabalho utiliza ainda uma forma aproximada de detecção "multi-vista" relacionando imagens aéreas e de vista panorâmica. Para considerar a contribuição da detecção multi-vista o trabalho propõe o uso de um modelo probabilístico *Conditional Random Field* (CRF) que permite integrar informações de diversas fontes ao modelo, como as imagens multi-vista, mas também informações como distância entre árvores e estradas, entre outras informações. Assim o uso desta técnica torna o sistema de detecção mais robusto, minimizando a desvantagem em relação a outras metodologias que utilizam mais fontes de dados, como LiDAR.

O processo de classificação de espécies de árvores é descrito abaixo.

A partir das localizações, coordenadas geográficas, das árvores detectadas é feito em seguida o *download* das imagens aéreas destas localizações, assim como o *download* de 3 imagens adicionais de vista panorâmica com diferentes níveis de *zooms*. O uso deste conjunto de imagens é útil pois permite extrair maiores informações sobre as espécies. Isso porque permitem retratar as árvores em diferentes níveis de detalhes, desde a forma da árvore e de galhos, num nível de *zoom* mais baixo, até textura do tronco e padrões existentes nas folhas, em níveis de *zoom* mais altos. Outra característica que é levada em conta, é um ângulo de visão chamado *pitch*. Isso porque existe uma inclinação adequada para a captura das imagens pelo GSV. Definidas estas características, para o aprendizado e tarefa de classificação, são usadas 4 CNNS do tipo VGG16. Estas são treinadas separadamente para cada um dos 4 tipos de imagem, ou seja, 1 imagem aérea e 3 imagens de vista panorâmica com níveis de *zoom* diferentes, para que assim possam extrair as características específicas. O treinamento das redes utilizou uma função custo log-logística e como algorítmo de otimização o algoritmo de descida de gradiente estocástico.

Para avaliação dos resultados primeiramente foi feita uma divisão do conjunto de dados em 70% para treino, 20% para validação e 10% para testes. Em segundo lugar, as diferentes tarefas presentes (detecção de árvores e reconhecimento de espécies) foram avaliadas de formas diferentes.

Para a detecção foram usados parâmetros de precisão e *recall*, sendo inclusive traçadas curvas precisão x *recall* varrendo diferentes *thresholds* do *score* de detecção. Em termos de acurácia de posicionamento, foi admitido que uma detecção em um raio de 4m no entorno de um valor de referência (*ground-truth*) é verdadeiro positivo. Os autores consideram esta uma hipótese razoável dado que bases de dados de árvores em geral ainda não apresentarem valores precisos desta informação. De qualquer modo, o aspecto mais importante que deve ser levado em conta é a capacidade de distinção entre 2 árvores vizinhas, a respeito disso, a maioria das detecções realizadas no trabalho, então num raio de 2m no entorno de valores de referência, dando um nível de confiabilidade para a hipótese.

Para o reconhecimento de espécies foram apresentadas medidas de precisão de *dataset* e média das precisões de classe. Na precisão de *dataset* é utilizada a taxa global de acertos ao longo de todo o conjunto de testes. Isso é feito independentemente do número de ocorrências por espécie. Assim a medida acaba sofrendo uma influência maior de espécies mais frequentes. Por outro lado, na média das precisões de classe é avaliada a precisão separadamente por espécie e após isso calculada a média dos valores de precisão obtidos para cada classe. Deste modo, esta medida permite avaliar os resultados mesmo para espécies menos frequentes, pois o valor de precisão de cada uma das classes tem mesma importância no cálculo.

3 Fundamentação Teórica

3.1 Aprendizado de Máquina

De forma geral, um processo básico de Aprendizado de Máquina consiste das seguintes etapas: Aquisição de dados, preparação (ou pré-processamento) de dados, definição de modelo que será usado em um problema específico de aprendizado de máquina, divisão de dados em dados de teste, de treino e de validação, aplicação dos dados ao modelo para realização do treino (processo de aprendizado), aplicação dos dados dados para avaliação do modelo obtido (validação), e finalmente aplicação dos dados em testes.

A respeito das principais aplicações existentes na área de aprendizado de máquina, é possível fazer uma divisão em tarefas preditivas e tarefas descritivas. No caso de tarefas preditivas, destacam-se os problemas de Classificação e Regressão, enquanto que para tarefas descritivas, é válido considerar o Agrupamento e Associação.

Primeiramente, a classificação é o processo de identificar o grupo ao qual pertencem dados com determinadas características. No caso da regressão, diz respeito à previsão de um determinado comportamento, para um conjunto de dados com parâmetros ou características numéricas. Para o agrupamento, de forma semelhante à classificação, devem ser reunidos dados que apresentem determinadas características, no entanto, neste caso não existem grupos, ou classes, previamente definidos. Por fim, na associação, o objetivo é encontrar quais parâmetros estão relacionados, assim determinar as regras que formam um determinado grupo.

3.1.1 Algoritmos de Aprendizado

A respeito do processo de aprendizado, existem 3 tipos principais de algoritmos [10] Aprendizado Supervisionado, Não-Supervisionado e Aprendizado por Reforço.

Aprendizado Supervisionado: É o caso normalmente utilizado nos problemas de classificação e regressão. Assim, o algoritmo utiliza como entradas, dados que já apresentam um rótulo ou uma estimativa. A partir de características presentes nestes dados, considerando que têm rótulos ou estimativas corretas, o algoritmo aprende, podendo fornecer classificações ou estimativas corretas para novas entradas.

Aprendizado Não-Supervionado: É o caso utilizado no problema de agrupamento, onde não é dada uma informação prévia das classes. Portanto, para este tipo de aprendizado as saídas são definidas a partir da observação de características presentes nas entradas independentemente dos resultados produzidos, que serão definidos exclusivamente pelo

processamento realizado.

Aprendizado por Reforço: É baseado na interação de um agente com o meio, utilizando ações de tentativa e erro, para as ações é aplicado um sistema de recompensas que define políticas para as futuras ações do agente.

3.1.2 Overfitting e Separação de Dados

No contexto de aprendizado de máquina, um algoritmo deve inicialmente aprender, para que, posteriormente, após estar capacitado para executar alguma tarefa, possa ser aplicado nesta tarefa. Deste modo, uma avaliação do desempenho dos algoritmos (validação), é interessante antes que sejam aplicados.

Considerando a etapa de avaliação do algoritmo, uma situação indesejada é que as entradas utilizadas pertençam ao conjunto de testes. Isso porque, em algum sentido, o algoritmo já tem um conhecimento prévio desta entrada, portanto, poderá ter um desempenho melhor, o que não é adequado considerando que, em situações reais, não terá esta vantagem. Outro problema ocorre caso não seja realizada a validação, ou seja, se todo conjunto de entradas for utilizado na etapa de treino. Neste caso não teríamos conhecimento sobre o desempenho do algoritmo em dados reais. Pode ainda ocorrer a situação em que o algoritmo apresenta bom desempenho em treino, no entanto, um desempenho ruim ao ser testado, isso porque o modelo está ajustado demais ao conjunto de dados, não conseguindo ser aplicado em dados diferentes daqueles em que foi treinado, pois não tem capacidade de generalização. Esta condição é chamada de *Overfitting*. Por fim, também deve ser observado que quando é feita esta separação, entre dados de treino e validação, estamos diminuindo o conjunto de treino, de modo que a etapa de aprendizado será menor, podendo diminuir o desempenho do algoritmo [11, 12].

Assim, verifica-se que deve haver uma separação do conjunto de entradas em um conjunto para treino e outro para realizar a validação. No entanto, este processo deve ser pensado com algum cuidado, visto que, existe um conflito de interesse na divisão entre os dados para treino e dados para teste.

3.1.3 Métricas de Avaliação

A seguir serão apresentadas algumas das principais métricas em problemas de aprendizado de máquina conforme [12].

3.1.3.1 Matriz de confusão e Métricas Associadas

Matriz de confusão é uma tabela que descreve o desempenho de um modelo de classificação binária. Seu uso é interessante particularmente quando estamos lidando com classes desbalanceadas, onde a frequência de ocorrência das classes em estudo é diferente.

Na sua construção a identificação das linhas corresponde às classes obtidas através do modelo proposto, enquanto que a identificação das colunas se refere a valores tidos como verdadeiros (referências). Finalmente, o preenchimento da tabela é feito com uma divisão dos resultados do processo de classificação em:

- Verdadeiros positivos (TP): São classificações corretas feitas para a classe de interesse da situação analisada.
- Verdadeiros negativos (TN): São classificações corretas feitas para a classe que não é de interesse da situação analisada.
- Falsos positivos (FP): São classificações incorretas feitas para a classe de interesse da situação analisada.
- Falsos negativos (FN): São classificações incorretas feitas para a classe que não é de interesse da situação analisada.

Tendo sido definida a matriz de confusão, é possível definir as seguintes métricas:

Acurácia: É a proporção de predições corretas (TP+TN) em relação ao total de predições (TP+TN+FP+FN).

Precisão: É a proporção de predições, em relação a classe de interesse (TP+FN), que são corretas (TP).

Revocação (Recall): É a capacidade de identificação das predições corretas. Ou seja, quantas predições corretas o modelo consegue realizar (TP) no total de predições (TP+TN+FP+FN).

Vale observar ainda que a diagonal principal da matriz de confusão exibe onde o classificador acertou. Enquanto que a diagonal secundária exibe onde o classificador errou.

Limiar de Classificação: será a regra que definirá a separação das classes do modelo, sendo, em geral, definido como valor de probabilidade. No entanto, o limiar é uma característica que muda de acordo com a situação estudada. Deste modo, conforme este limiar é modificado, os resultados da classificação podem mudar, e portanto as métricas anteriormente definidas também mudam.

Sensibilidade: é a proporção de verdadeiros positivos presentes na classe positiva. Assim, conforme este valor aumenta a classificação para a classe complementar a de interesse se torna mais confiável. Isso porque, aumentar a sensibilidade corresponde a diminuir a proporção de falsos negativos, e isto corresponde a aumentar a precisão da classificação negativa.

Especificidade: é a proporção de verdadeiros negativos presentes na classe negativa. De forma análoga à sensibilidade, verifica-se que uma alta especificidade corresponde ao aumento da precisão da classificação positiva.

Curva de Característica de Operação do Receptor (ROC, do inglês Receiver Operating Characteristic Curve): É um gráfico mostrando a performance de modelos de classificação em diferentes limiares. As medidas de desempenho usadas são a sensibilidade (ou taxa de verdadeiros positivos), em função do complementar da especificidade (ou complementar da taxa de falsos positivos). Assim, são mostradas estas duas medidas conforme é alterado o limiar de classificação com diferentes valores, formando portanto a curva.

Curva de Precisão-Recall: Como apresentado, 2 métricas de interesse em problemas de classificação são a precisão e recall. No entanto, existe um balanço entre as duas métricas, visto que, conforme o valor de recall aumenta, o valor de precisão diminui, o que é esperado, visto que, aumentando a quantidade de amostras de uma classe, aumenta-se o recall, a quantidade de acertos para a classe de interesse, mas diminui-se a frequência de classificações corretas em relação a esta classe. Portanto, a Precisão-Recall apresenta o relacionamento entre ambas as métricas, permitindo assim definir a contribuição de cada uma na avaliação do modelo.

3.1.3.2 Função Custo

Nos problemas de aprendizado, é interessante que se tenha uma medida que possa indicar o quão próximos estão os resultados ou saídas fornecidas pelo algoritmo do desempenho desejado. Assim, para representar esta medida, a função custo é definida por alguma medida que quantifique a diferença existente entre saídas do algoritmo e valores desejados. Tendo definido uma função custo, ela terá utilidade na definição dos métodos de aprendizado que poderão ser definidos buscando minimizar esta medida. Existem diferentes definições de função custo. No caso de problemas de classificação binária e de muitas classes é utilizada como função custo a Entropia Cruzada. Para problemas de regressão pode ser utilizada como função custo o Erro Quadrático Médio, que intuitivamente pode ser definido como o valor mínimo de erro presente no ajuste de uma curva a um conjunto de pontos conhecidos [8].

3.1.3.2.1 Entropia Cruzada e Máxima Verossimilhança

Uma Função de Verossimilhança define uma estimativa sobre a probabilidade de uma determinada observação utilizando a hipótese de que a observação segue uma determinada distribuição de probabilidade. Neste sentido, o critério de máxima verossimilhança faz uma estimativa sobre um conjunto de observações considerando a hipótese de que estas observações sigam certa distribuição de probabilidade. A forma de realizar a estimativa do

conjunto de pontos é através do produtório de seus valores, mas é comumente definida em termos de logaritmo, por aspectos práticos.

A entropia cruzada entre duas distribuições de probabilidade p e q pode ser definida como a soma da entropia da distribuição p e a entropia relativa entre p e q. Verifica-se que a minimização da entropia cruzada é equivalente à maximização de verossimilhança. Deste modo, fazer a minimização de entropia cruzada é interessante para realizar estimativas corretas das amostras, e portanto realizar classificações corretas [8].

3.2 Redes Neurais

Redes Neurais consistem de um conjunto de métodos de aprendizagem não lineares (portanto, que podem ser aplicados a problemas não linearmente separáveis), com uma representação baseada em grafos. As redes neurais são inspiradas no comportamento simplificado de neurônios biológicos onde estímulos de entrada passam por um processamento produzindo estímulos na saída. O objetivo é ajustar os parâmetros da rede neural para definir uma estrutura que consiga modelar adequadamente um problema de interesse, podendo prever o comportamento de uma ou mais saídas para um conjunto de de entradas.

Para formalização do modelo de rede neural, primeiramente é definido o modelo de um neurônio artificial, chamado perceptron. O modelo matemático de um *perceptron* é composto de 3 elementos básicos: combinação linear das entradas ponderadas pelos pesos sinápticos associados a cada entrada do neurônio, viés e função de ativação. A combinação linear das entradas adicionada ao limiar pode ser representada pela equação 3.1, a função de ativação será aplicada sobre este valor:

$$z = \sum_{u=1}^{m} w_i x_i + b \tag{3.1}$$

Onde $\sum_{u=1}^{m} w_i x_i$ é a combinação linear das entradas e b o valor de limiar.

Neste sentido, o modelo básico de uma rede neural é definido pela conexão de unidades, também chamadas neurônios, e que seguem formulação similar a do perceptron, em multiplas camadas. Demonstra-se que é possível obter uma aproximação adequada para qualquer função que seja suave através de uma combinação linear das entradas e aplicando uma função não-linear. Portanto, o modelo de perceptron multicamada, ou seja, uma rede neural, pode ser aplicada como aproximador universal de funções, conseguindo assim modelar diversos problemas de interesse [10, 11].

3.2.1 Função de ativação

As funções de ativação, também chamadas funções de transferência, têm como principal objetivo introduzir não-linearidade ao modelo de redes neurais, o que é feito através do mapeamento de entradas para saídas desejadas aos problemas de interesse (regressão, classificação, entre outros). Por outro lado, a escolha de funções de ativação, de acordo com a estrutura de rede neural que está sendo utilizada, tem implicações nos parâmetros de desempenho dos algoritmos.

O tipo mais simples de função de ativação não-linear é a função degrau, definida pela equação 3.2 a seguir:

$$\begin{cases} f(z) = 1, \text{se } z >= 0 \\ f(z) = 0, \text{se } z < 0 \end{cases} \tag{3.2}$$

Observa-se que a função de ativação degrau atua como classificador binário, onde a classificação é feita pela comparação do valor aplicado com um valor limiar. A princípio este limiar é zero, no entanto, como o valor aplicado é composto pela combinação linear das entradas adicionada ao valor limiar, podemos comparar a combinação linear das entradas com um valor de limiar definido pelo valor limiar. Além disso, como o perceptron simples define um classificador binário, então é aplicado apenas a problemas linearmente separáveis, onde a classificação pode ser feita com uma única fronteira de decisão em linha reta. Assim, é a conexão de multiplas camadas de perceptrons que aumenta a capacidade de representação do modelo, permitindo que seja aplicado a problemas não-lineares.

Será verificado que no processo de aprendizado feito em redes neurais a função de ativação passa por operações de diferenciação, portanto, sendo necessário que sejam utilizadas funções diferenciáveis. Assim, o uso da função degrau se torna inadeqado, uma alternativa é utilizar uma função que realize um mapeamento semelhante ao degrau, de entradas em classes 0 e 1, no entanto com característica mais suave, ou seja, com uma função diferenciável. Assim, são propostas funções como sigmóide e tangente hiperbólica [10, 11]. A título de exemplo, pode-se apresentar a função sigmóide , que realiza o mapeamento das entradas para o intervalo de valores reais entre 0 e 1. Neste caso os valores representam probabilidades, e o mapeamento das classes é feito de acordo com o critério de decisão pela maior probabilidade, onde, sendo duas classes equiprováveis, será utilizado um valor limiar de 0.5. A a função sigmóide é descrita pela equação 3.3

$$f(z) = \frac{1}{1 + e^{-z}} \tag{3.3}$$

Uma generalização para a função de ativação sigmóide é a função de ativação [**?**], que é aplicada em problemas de classificação multiclasse. Neste caso também é realizada a decisão pela classe com o maior valor de probabilidade. Além disso, a função [**?**] se baseia

na normalização dos valores das probabilidades das classes para que respeitem os axiomas da teoria de probabilidade [8].

3.2.2 Aprendizado em Redes Neurais

Foi observado que redes neurais atuam como aproximadores universais de funções, sendo que, através de seus parâmetros (pesos sinápticos e limiares) é possível ajustar a aproximação feita. De modo geral estes parâmetros são inicializados de forma aleatória, e em seguida é realizado o processo de aprendizado onde é feita a atualização destes parâmetros visando obter uma melhor aproximação para a função de interesse. Lembrando que a função custo indica quão boa é a aproximação, então a otimização dos parâmetros é realizada visando a minimização da função custo [11, 13].

A condição para minimização da função custo em relação aos pesos da rede é dada pela equação 3.4 a seguir:

$$\frac{\partial J}{\partial w_i} = 0, i = 0, 1, ...n - 1 \tag{3.4}$$

A partir da equação 3.4 é definido o conjunto de pesos ótimos que minimizam a função custo. Assim, de modo geral, é possível obter uma solução analítica para os pesos. No entanto, comumente eles são ajustados iterativamente através da equação 3.5:

$$w_{i+1} = w_i - \mu \nabla J(w) \tag{3.5}$$

Onde w_i é o vetor de pesos para a iteração i, μ é a taxa de aprendizado e $\nabla J(w)$ é o gradiente da função custo J, que é função do conjunto de pesos.

Observa-se que o ajuste é feito na direção do vetor $-\nabla J(w)$, correspondendo à direção de decrescimento da função custo. Vale observar que podem haver variações no algoritmo de otimização. Alguns exemplos conhecidos são o "Stochastic Gradient Descent"que realiza o cálculo do gradiente de forma estocástica, e o Adaptive Momentum Estimation que utiliza estatísticas adicionais como primeiro e segundo momento dos gradientes [14].

3.3 Deep Learning

Uma forma de simplificar a resolução de determinados problemas é através da identificação e interpretação das características que melhor os representam. No entanto, algumas vezes pode ser difícil saber quais características estão envolvidas. Assim é introduzido o conceito de aprendizagem de representações, onde o objetivo é usar aprendizado de máquina para determinar a melhor representação para o conjunto de dados do problema.

Paralelamente, uma forma de obter uma melhor representação para o conjunto de dados é através da extração das características mais significativas, e identificação de fatores de variabilidade que estejam presentes com remoção daqueles que sejam indesejáveis. No entanto, para alguns tipos de problemas pode ser difícil realizar a extração de características e remoção de fatores de variabilidade presentes, devido à alta complexidade e alto nível de abstração requerido. Nestes casos, uma abordagem possível é a utilização de características compostas, ou seja, que possam ser expressas em termos de outras características. Esta é a ideia principal presente em Deep Learning, e para isso uma das formas de determinar estas características é fazendo a decomposição do mapeamento existente entre o conjunto de dados e a representação desejada em mapeamentos mais simples. Os mapeamentos estarão em um conjunto de camadas, e nestas camadas estarão disponíveis as características mais simples, que formam, através de composições a partir dos níveis mais baixos, a representação do problema.

O uso de redes neurais multicamadas permite realizar a composição de características e representações através de suas camadas. Isso é possível pois uma rede neural multicamada (multilayer perceptron, MLP) é formada pela composição de funções matemáticas simples. Assim é um método adequado para as tarefas de Deep Learning [8].

3.4 Redes Neurais Convolucionais

A área de processamento de imagens e visão computacional se caracteriza por métodos computacionais para interpretação e extração de informações obtidas por sensores visuais. Existem algumas técnicas bem consolidadas que caracterizam a visão computacional clássica. Neste conjunto de técnicas procura-se representar e manipular informações provenientes de características fortemente relacionadas com a capacidade humana de interpretação visual. Deste modo são apresentados métodos baseados em atributos como intensidade de cor/brilho, textura, formas presentes em objetos da imagem e vizinhança. [2, 15]

O uso de técnicas de aprendizado de máquina como “Máquinas de Vetor Suporte”, “Vizinhos Mais Próximos”, “Análise de Componentes Conectadas” e “Redes Neurais” já era utilizado permitindo auxiliar a resolução de problemas nestas abordagens. No entanto, recentemente tem sido verificada a eficácia de métodos puramente baseados em redes neurais para a resolução destes problemas, em especial métodos de Deep Learning [7], o que está de acordo com 3.3.

Embora a aplicação de redes neurais multicamadas seja possível, o fato de serem densamente conectadas (todos neurônios de camadas adjacentes estão conectados) leva ao problema de aumento significativo no número de pesos conforme aumentamos o tamanho da imagem, dificultando o uso de um número maior de camadas para extração das

características. Além disso, MLPs não fazem distinção nem aproveitam informações espaciais presentes no relacionamento entre pixels da imagem. Este contexto serve como uma intuição para a aplicação das *Convolutional Neural Network* (CNNs), considerando que para imagens existe relacionamento espacial entre os pixels que pode ser explorado. No entanto, a origem das CNNs é baseada mais diretamente em inspiração biológica, em especial nas pesquisas de Hubel e Wiesel que mostraram o importante papel do córtex visual primário (V1) no processamento de sinais visuais. Assim, as CNNs foram projetadas para conseguir reproduzir propriedades do V1. [8, 16, 17]

Em síntese pode-se que dizer que o V1 é caracterizado como uma estrutura bi-dimensional (um mapa representando informações em um espaço de 2 dimensões) que reflete os estímulos aplicados à retina. Além disso, possui células simples (small cells) e células complexas (complex cells). As células simples definem um campo receptivo local, isso porque apresentam grandes restrições a modificações nos estímulos recebidos, com isso, a distribuição ou deslocamentos espaciais de estímulos também estarão limitados. Por outro lado, células complexas definem um campo receptivo maior. Mais precisamente, Hubel e Wiesel chegaram a conclusões de que células complexas recebem estímulos de um conjunto de células simples que possuem os mesmos tipos de restrições, como as restrições são as mesmas, a seletividade de células complexas é menor, correspondendo a respostas mais invariantes à translação, ou deslocamento. [8, 16]

Resumidamente, CNNs são definidas por uma arquitetura contendo camadas de convolução intercaladas por camadas da operação de *Pooling*, seguidas finalmente por um classificador baseado em uma MLP. Neste sentido, percebe-se que células simples que podem ser caracterizadas por operações locais e lineares em resposta aos estímulos recebidos, correspondem às camadas de convolução. Por outro lado, células complexas, que correspondem à combinação de estímulos recebidos, podem ser representadas através das camadas e operações de *Pooling*. [8]

A operação de convolução espacial, é caracterizada pela equação:

$$B(i,j) = K \circledast A = \sum_{u=-k}^{k} \sum_{v=-k}^{k} A(i-u, j-v)K(u,v) \tag{3.6}$$

Onde K se refere ao filtro (ou kernel) e A se refere à imagem de entrada.

Assim, as saídas correspondem a combinações lineares de pixels vizinhos ponderados pelos valores dos pesos do filtro. No contexto de CNNs o conjunto de saídas da operação de convolução costuma ser chamado de mapa de características.

No entanto, as implementações de CNNs em geral não utilizam a operação de convolução, ao invés disso, utilizam correlação cruzada, onde o sentido do kernel não é invertido em relação às entradas (imagem), o que, no entanto, não influencia os resultados.

Deste modo a equação aplicada é:

$$B(i,j) = K \circledast A = \sum_{u=-k}^{k} \sum_{v=-k}^{k} K(u,v)A(i+u, j+v) \tag{3.7}$$

Além do tamanho e pesos do filtro, outros parâmetros considerados nas operações de convolução são o *padding* e o *stride*. O *padding* corresponde a um preenchimento (por zeros) aplicado à entrada levando a uma correção do tamanho da saída para que sejam mantidas dimensões de interesse. Se o interesse é que seja mantida a dimensão da saída, então o padding não deve ser adicionado (deve ter tamanho nulo), enquanto que, se houver interesse em alterar o tamanho da saída, é aplicado um padding que preencha lacunas presentes. Por outro lado, o stride é o tamanho dos passos que serão usados no deslocamento do filtro para percorrer a imagem, portanto consiste dos valores i e j na equação de convolução. Observe que o tamanho do passo influencia a resolução da imagem de saída, isso porque, se o filtro de convolução percorre a imagem em passos maiores, produz saídas, ou seja, mapas de características, menores. [8, 18]

O relacionamento entre o tamanho de padding e a imagem de saída é descrito de forma mais precisa através da equação:

$$w_s = (\frac{w_i + 2p - k}{s}) + 1 \tag{3.8}$$

Onde w_s é o tamanho da saída, w_i o tamanho da imagem, 'p' o tamanho do padding, 'k' tamanho do filtro e 's' tamanho do stride.

As camadas de pooling estarão situadas logo a seguir das camadas de convolução. A operação de pooling realiza uma condensação das saídas da camada de convolução, ou seja, realiza uma simplificação e redução de dimensionalidade. [17]

O uso das camadas de convolução e de pooling consegue extrair informações espaciais e corrigir a ineficiência presente nas MLPs pelo aumento significativo do número de pesos. Isso porque diminui o número de pesos se forem satisfeitas duas premissas, conectividade esparsa e compartilhamento de pesos. Mais precisamente, na operação de convolução a aplicação dos pesos às entradas estará limitada pelo tamanho do filtro, definindo conectividade esparsa. Além disso, como os mesmos pesos do filtro são aplicados para cada região da imagem conforme esta é percorrida no processo de convolução, então, também existe compartilhamento de pesos. Assim, a arquitetura, inspirada na estrutura biológica do córtex visual consegue resolver os principais problemas existentes para a aplicação de MLPs no problema de processamento de imagens. [8, 18]

3.4.1 Função de ativação ReLU

Um dos processos presentes ao realizar o treinamento de uma RN é a atualização de seu modelo (os pesos da rede) através da retropropagação do gradiente da função custo que ocorre camada por camada das saídas para as entradas. No entanto, dois problemas que podem ser encontrados ao realizar o treinamento de redes neurais, especificamente no caso de redes de muitas camadas são os de explosão ou desvanecimento de gradiente da função custo. De maneira simplificada isso ocorre porque conforme é feita a propagação entre as diferentes camadas ocorre multiplicação dos pesos das camadas. Assim, se a magnitude dos pesos é pequena, devida a grande quantidade de camadas ocorrerá desvanecimento, enquanto que, se a magnitude for grande, poderá haver a explosão (divergência) dos valores.

Considerando estes problemas, é conhecido que o uso da função de ativação *Rectified Linear Unit* (ReLU) ajuda a evitá-los. Isso, porque, a forma como é proposta permite que ao aumentar o peso das entradas ponderadas elas não sejam saturadas, enquanto que, se o valor ponderado é negativo os valores são descartados. Por este motivo a função é amplamente utilizada como função de ativação em arquiteturas profundas. [17, 18]

3.4.2 Regularização e Droput

Como apresentado em 3.1.2, um dos problemas presentes em algoritmos de aprendizado de máquina é a ocorrência de *overfitting*, ocorrendo quando o modelo está demasiadamente ajustado ao conjunto de dados de treinamento não conseguindo ser aplicado a dados diferentes por não possuir capacidade de generalização. As técnicas aplicadas para corrigir este problema são chamadas técnicas de regularização. Uma técnica de regularização bastante utilizada em modelos de *Deep Learning* é o uso de *Dropout*. Nesta técnica parte dos neurônios de uma determinada camada da rede neural é desconsiderada a cada iteração no treino. Isso permite que a rede seja treinada com diferentes arquiteturas, reduzindo o *overfitting* e aumentando capacidade de generalização. [8]

3.4.3 Data augmentation

De modo geral em algoritmos de aprendizagem de máquina para aumentar a capacidade de generalização dos algoritmos, é necessário utilizar um conjunto de dados maior. Neste sentido, em situações em que pode ser difícil obter maiores quantidades de dados, uma possibilidade é produzir novos dados através de transformações no conjunto de dados original, este procedimento de geração de novos dados é a aumentação de dados (*Data Augmentation*).

Deve ser observado que nem todos problemas de aprendizado de máquina permitem que seja aplicada esta abordagem. No entanto, em tarefas de classificação em processamento

de imagens, e em tarefas de detecção de objetos e classificação de imagens isso é possível. Nestes casos, podem ser aplicadas transformações como rotação, escala, modificação de brilho, modificação de tamanho, etc. [8, 18]

3.4.4 Batch normalization

Uma característica desejável em algoritmos de aprendizado de máquina é que os dados das entradas sejam normalizados. Isso porque dados com escalas diferentes podem influenciar de maneira diferente o desempenho e aprendizado do algoritmo que pode dar uma importância maior para valores com escalas maiores.

A normalização em batch pode ser aplicada a qualquer tipo de camada, entretanto, no caso de redes convolucionais, é interessante que seja aplicada entre camadas ocultas, para que os valores passados estejam normalizados. Isso porque realizar transformações e processamentos nestas camadas pode ter influências inesperadas e indesejáveis na aplicação da função de ativação. Deste modo, a normalização em *batch* faz uma transformação das entradas corrigindo este problema, e alcançando o objetivo inicial de aumentar a estabilidade da convergência do algoritmo. Outro resultado é que a normalização em *batch* também tem um efeito de regularização. Deste modo, quando aplicada não são necessários outros métodos como o *dropout*. [18, 19]

3.4.5 Transferência de Aprendizado e Ajuste Fino

A transferência de aprendizado é um procedimento utilizado em implementações de aprendizado de máquina que visa aproveitar modelos previamente treinados em uma determinada tarefa ou configuração aplicando-os em uma situação diferente, mas que utilize de características em geral parecidas. Para o aprendizado de características visuais, podemos aproveitar modelos que tenham sido treinados e conseguem reconhecer determinados atributos em imagens de domínios razoavelmente diferentes. Esta técnica pode ser interessante por garantir capacidade de generalização ao domínio de interesse ou em casos em que o dataset seja razoavelmente pequeno sendo difícil realizar o treinamento de um modelo por completo. [8]

A aplicação de transferência de aprendizado no contexto de DL é feita através da aplicação do modelo de rede convolucional, previamente treinado, como modelo para o problema de interesse. Assim as camadas iniciais do modelo que será utilizado correspondem às camadas do modelo previamente treinado. No entanto, para que o conhecimento proveniente destas camadas não seja perdido no seguinte treino, é feito o seu "congelamento", ou seja, os parâmetros da rede convolucional nestas camadas deixam de ser atualizados. São adicionadas finalmente camadas que possibilitem o aprendizado das características específicas do problema, apenas estas camadas são treinadas [20].

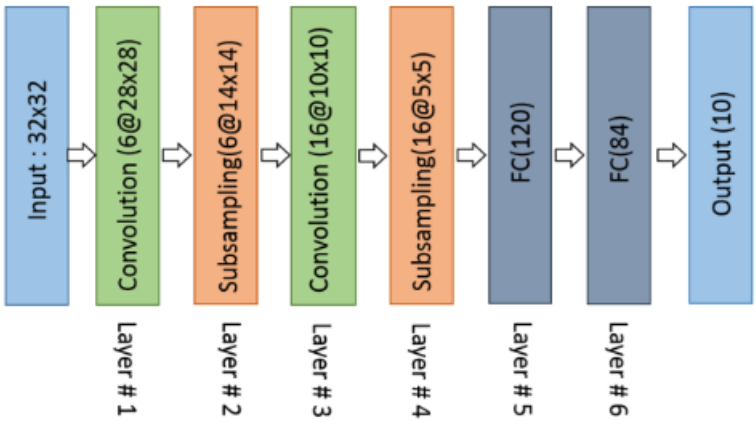
O ajusto fino é feito utilizando uma taxa de aprendizado pequena para que o modelo não seja significativamente alterado, e é aplicado principalmente em situações em que o modelo já foi treinado no dataset de interesse, mas precisa ser aplicado em casos com pequenas diferenças, ou também quando o modelo já apresenta resultados razoavelmente bons, visando pequenas melhorias. A aplicação do ajuste fino pode ser feita em um modelo que foi treinado por completo ou também para situações em que foi utilizada previamente transferência de aprendizagem, nesta condição o congelamento do modelo deixa de ser aplicado da forma como era feito, para que todo o modelo seja levemente ajustado [20].

3.4.6 Arquiteuras de Redes Neurais Convolucionais

Tendo apresentado alguns dos principais conceitos envolvidos com técnicas de DL e de CNNs, torna-se simples compreender como passaram a ser utilizadas as principais arquiteturas usadas atualmente. Assim, será apresentada uma intuição sobre algumas de grande importância e em especial as ques serão usadas neste trabalho.

Uma das primeiras arquiteturas de CNN propostas foi a LeNet, utilizando elementos como camadas convolucionais, camadas de pooling e camada de classificação com MLP [21, 18]. A arquitetura LeNet é mostrada na Figura 2 a seguir:

Figura 2 – Arquitetura da CNN LeNet

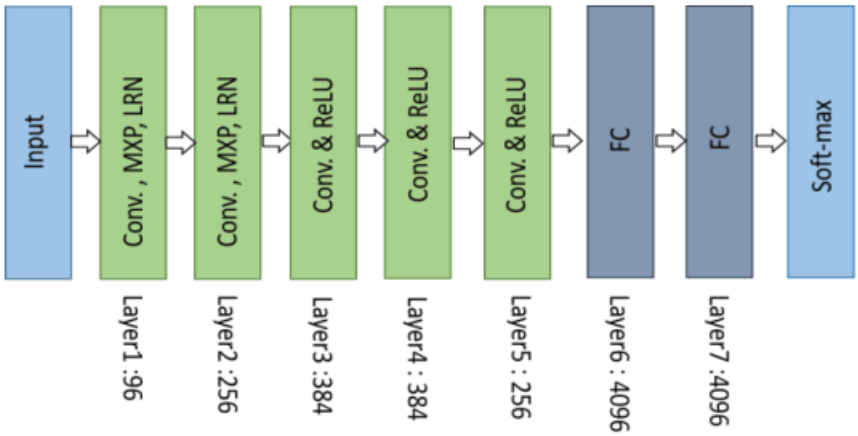


Fonte: Referência [21]

A arquitetura seguinte à LeNet foi a AlexNet, que passou utilizar a função de ativação ReLU, corrigindo os problemas de desvanecimento e explosão do gradiente e isso permitiu o aumento do número de camadas [21, 18], como é mostrado na Figura 3 a seguir:

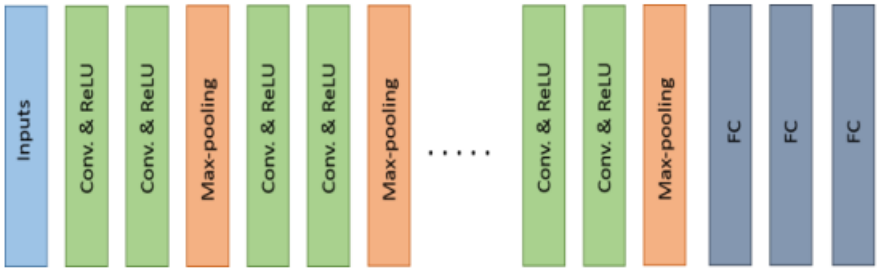
Seguindo a tendência da AlexNet de aumento do número de camadas é apresentada VGG16 que tem como proposta utilizar filtros de tamanho menor permitindo aumentar ainda mais o número de camadas[21, 18]. Diminuindo o tamanho dos filtros é possível diminuir o número de parâmetros (pesos dos neurônios). A arquitetura VGG16 é mostrada na Figura 4 a seguir:

Figura 3 – Arquitetura da CNN AlexNet



Fonte: Referência [21]

Figura 4 – Arquitetura de CNN VGG16

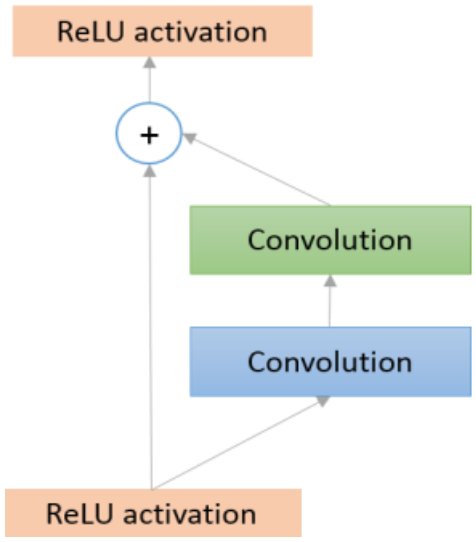


Fonte: Referência [21]

Após a VGG16 foi proposta a rede Inception (ou GoogleNet) utilizando um banco de filtros com tamanhos diferentes permitindo explorar diferentes informações presentes na imagem. Em seguida foi proposta a ResNet50, apresentando, como inovação, conexões residuais entre as camadas. O uso destas conexões permite que possam ser utilizados modelos mais profundos. Isso porque as conexões em salto conectam camadas de diferentes níveis, criando um caminho de informações diferente que consegue corrigir distorções que ocorrem num caminho com muitas camadas (como o desvanecimento ou explosão do gradiente). [21, 22]

A arquitetura ResNet50 é mostrada na figura 5 a seguir:

Figura 5 – Arquitetura de bloco elementar da CNN ResNet50



Fonte: Referência [21]

A respeito das arquiteturas consideradas, em especial nos trabalhos [7] e [1], deve-se observar que atualmente apenas parte delas ainda é utilizada. No caso da VGG16, que pode ser utilizada na tarefa de classificação e reconhecimento de imagens, verifica-se que não é adequado realizar o treinamento da rede, por apresentar uma quantidade de parâmetros grande em comparação com outras mais recentes [21], no entanto, seria possível utilizar uma versão pré-treinada com técnicas de transferência de aprendizado. Ao invés da VGG16, neste trabalho, será utilizada a ResNet que continua sendo uma boa solução para as tarefas de reconhecimento e classificação de imagens. Em relação a RetinaNet, embora existam outras arquiteturas (para a tarefa de detecção de objetos) com desempenho melhor, continua entre as melhores arquiteturas, além de apresentar relativa simplicidade para implementação e também será utilizada [21] .

3.5 Métricas para processamento de imagens

3.5.1 Índice de Jaccard

O índice de Jaccard (JI, do inglês, *Jaccard Index*), tem sido empregado para medir a similaridade entre conjuntos de amostras em diferentes problemas, sendo calculado pela cardinalidade da intersecção sobre a cardinalidade da união, entre estes conjuntos. Por este motivo também é conhecido como a medida de "intersecção sobre união"IoU.

Em visão computacional, a medida JI pode ser dada como área (em número de pixels) de sobreposição entre as imagens de referência (ground-truth) e imagens preditas. Na tarefa de detecção de objetos as imagens de referência são definidas como as áreas delimitadas pelas caixas delimitadoras, enquanto que na tarefa de segmentação as imagens

de referência são definidas pelas máscaras binárias. [23]

Finalmente, pode-se verificar [24] que o índice de Jaccard pode ser descrito em termos das métricas associadas à matriz de confusão. Sendo possível demonstrar que seu valor é dado pela equação 3.9:

$$JI = \frac{TP}{FP + TP + FN} \tag{3.9}$$

3.6 Detecção de Objetos

Um problema presente em visão computacional que difere da classificação é a detecção de objetos. Neste caso, o interesse está não apenas em determinar a classe de um objeto, mas adicionalmente, verificar se o objeto está presente na imagem e determinar a sua localização. Para lidar com este tipo de problema utilizando métodos de DL, pode-se considerar como intuição o caso de 1 objeto. Considerando isso, na localização de objetos a arquitetura deve utilizar um tipo diferente de entrada, onde as imagens são associadas não apenas a classes mas também a coordenadas e dimensões de elementos que fazem a localização do objeto na imagem, estes elementos correspondem a menor área (por padrão, com forma retangular) que envolve o objeto de interesse e são chamados caixas delimitadoras (*bounding boxes*). Além de caracteristicas específicas para as entradas, a arquitetura de rede neural para os detectores de objetos também é modificada visando aproveitar melhor este tipo de entrada e obter melhores desempenhos nas tarefas de interesse. Neste sentido, ainda é utilizada uma rede de camadas convolucionais, para extração de características das imagens, no entanto, a saída será dividida em dois módulos, uma rede MLP para a detecção de objetos, consistindo de um classificador, e outra rede MLP para regressão linear, que permitirá trabalhar com atributos numéricos como as coordenadas e dimensões das caixas delimitadoras. Finalmente, para realizar o treinamento desta arquitetura é utilizada uma função custo mista equilibrando a importância dos dois diferentes critérios. [25, 26]

Após apresentar uma intuição sobre como são definidas as arquiteturas de DL para detecção de objetos, deve-se considerar tarefas mais complexas como a detecção de multiplos objetos podendo utilizar multiplas classes. Para estas tarefas são utilizados modelos mais complexos que definem o estado da arte na detecção de objetos. Existem duas abordagens que podem ser utilizadas, modelos de 1 estágio, e modelos de 2 estágios que serão apresentados a seguir.

3.6.1 Modelos de 1 e 2 Estágios

A principal distinção presente entre os atuais modelos de detecção de objeto (de 1 ou 2 estágios) está na forma como é realizada a proposta de regiões com maior probabilidade

de conter objetos. Assim, nos modelos de 2 estágios, a proposta de regiões é feita em um primeiro estágio, utilizando algoritmos de visão computacional clássica que já vinham sendo utilizados nesta tarefa, como é feito, por exemplo, na arquitetura R-CNN. A proposta de regiões pode também ser feita utilizando uma rede neural para proposta de regiões (Region Proposal Network, RPN), como é feito na arquitetura Faster R-CNN. Assim, nas arquiteturas de 2 estágios, temos em seguida um estágio para a classificação e detecção. Por outro, os modelos de 1 estágio, conseguem realizar simultaneamente a proposta de regiões e a classificação. Diferente dos modelos de 2 estágios, todas as arquiteturas de 1 estágio são baseadas inteiramente em RNs. [25, 26]

Para o detalhamento de modelos de 2 estágios, pode-se considerar como exemplo a arquitetura R-CNN, que utiliza o algoritmo "Selective Search"para propor regiões de interesse (ou seja, caixas delimitadoras) através da divisão e posterior agrupamento de elementos da imagem baseando-se em diversas características visuais (cor, textura, fechamento, etc). Após a definição das regiões propostas, o resultado é passado para uma rede de camadas convolucionais produzindo um mapa de características que será passado para um classificador do tipo máquina de vetor suporte. O classificador realiza a classificação binária para as diferentes classes de forma independente baseando-se no valor limiar do índice IoU. Por fim, observa-se que o número de regiões propostas, que é definido pelos parâmetros do modelo, é um valor elevado, muito maior do que a quantidade de objetos presentes na imagem. Deste modo, embora a quantidade de regiões propostas seja reduzida ao fazer a limiariação das diferentes classes através do índice IoU, ainda é necessário um estágio seguinte para reduzir o número de regiões, este estágio é chamado *Non Maximum Supression* (NMS) onde é mantida a região com o maior valor para a métrica de avaliação utilizada no modelo de classificação. [27, 26]

Para o detalhamento de modelos de 1 estágio, será considerada a arquitetura RetinaNet, sendo esta uma das arquiteturas utilizadas neste trabalho. Neste sentido, como já observado, deve-se considerar que em modelos de 1 estágio as etapas de proposta de regiões e classificação são realizadas simultaneamente. Esta modificação é possibilitada pela introdução do conceito de âncoras. Nos modelos de 2 estágios a cobertura do espaço de caixas delimitadoras para as imagens era feita pelos algoritmos de proposta de região seguidos por uma operação de *Pooling* para regiões de interesse produzindo um mapa de características fixo para as diversas entradas geradas pela proposta de regiões. Por outro lado, nos modelos de 1 estágio a cobertura do espaço de caixas delimitadoras é realizada através uma grade fixa, assim para alcançar a cobertura são definidas âncoras. Âncoras são conjuntos de caixas delimitadoras pré-definidas através de diferentes valores de escala e razão de aspecto, estando associadas a cada um dos pontos do mapa de características. Assim, através das âncoras é possível fazer a cobertura do espaço de caixas delimitadoras, sendo portanto possível realizar a detecção através de um relacionamento entre as saídas do modelo (ou seja, caixas delimitadoras para os objetos de interesse) e as âncoras, que

funcionam como referências para estas. Portanto não existe necessidade da proposta de regiões, basta fazer a classificação e ajuste de offsets das saídas produzidas em relação às âncoras dos objetos de interesse. [25]

Uma segunda característica que tem sido adotada pelos modelos de 1 estágio, assim como outros modelos e aplicações de processamento de imagens com DL, são as *Feature Pyramid Networks* (FPN). Métodos clássicos de visão computacional já utilizavam pirâmides de imagens nas tarefas de detecção de objetos em multipla escala, no entanto, com problemas de alto custo computacional. Assim, implementações iniciais de DL não utilizaram esta característica. Por outro lado, percebeu-se que, ao invés do uso de piramides de imagens, seria possível explorar informações multi-escala utilizando as saídas de diversos estágios de uma rede convolucional, pois a estrutura hierarquica das redes convolucionais naturalmente realiza a extração de características, com isso foi apresentada como alternativa as *Pyramid Feature Hierarchy* (PFH). No entanto, PFH's têm como problema o fato de camadas com maior resolução terem informação semântica menor. As FPN's conseguem contornar este problema, pois, embora também utilizem saídas de diferentes estágios, a informação de alto nível é propagada para todas as camadas da pirâmide. Isso é possível pois utilizam 2 fluxos de dados, *bottom-up* e *top-down*. O sentido *bottom-up* é o sentido convencional das redes convolucionais que realiza a extração de características com aumento do valor semântico. Por outro lado, no sentido *top-down* é aumentada a resolução, mas o valor semântico diminui. [25, 28]

A construção das FPN's, conforme proposta em [28], tem particularidades adicionais que devem ser consideradas. Primeiramente, considera-se que podem ser construídas utilizando uma arquitetura genérica de rede convolucional, por exemplo, a ResNet. Em segundo lugar, considerando o fluxo no sentido bottom-up, deve-se observar que a cada um dos diferentes níveis da pirâmide corresponde um conjuntos de camadas. Isso porque determinadas camadas não alteram o tamanho/escala do mapa de características. No entanto, existe aumento do valor semântico entre as camadas de um estágio. Deste modo, é convencionado como saída, que será propagada para o próximo nível, o mapa de características da ultima camada do nível, pois é o mapa de características com maior valor semântico. A redução de tamanho entre níveis é dada pelo uso de diferentes razões (4, 8, 16, 32) do *stride* em relação ao tamanho da imagem de entrada. Enquanto que, no sentido *top-down*, a estrutura aumenta a resolução (*upsampling*) dos mapas de características através de uma convolução transposta, operação inversa ao processo de convolução. Finalmente, o fluxo se inicia nos níveis mais altos da pirâmide, os quais possuem alto valor semântico, assim, tendo aumentado a resolução destes mapas, eles são integrados a camada de mesmo nível. para a propagação de informação com alto valor semântico são utilizadas conexões residuais. Assim, iniciando-se pelos níveis mais altos da pirâmide, os quais possuem alto valor semântico, e tendo aumentado a resolução destes mapas, eles são integrados a camada de mesmo nível. [25, 28]

3.6.2 Função Custo Focal

É admitido que classes de fundo são mais frequentes do que as classes de objetos nos datasets para as tarefas de processamento de imagem. Este é um problema nas arquiteturas de detecção de 1 estágio, visto que não é feita uma pré-seleção das regiões analisadas. Neste sentido, a função custo focal, apresentada em [25], procura amenizar este problema. Para isso, procura dar prioridade às imagens com classes de objeto ao invés de classes de fundo que já possuem âncoras corretas.

Em termos quantitativos, incialmente é considerada a definição para a *Binary Cross-Entropy* (BCE) para uma classe de referência $y \in \{\pm 1\}$ com probabilidade $p \in [0, 1]$, esta é definida pelas equações 3.10 e 3.11:

$$\begin{cases} p_t = p, \text{se } y = 1 \\ p_t = 1 - p, \text{se } y = 0 \end{cases} \tag{3.10}$$

$$BCE(p, y) = BCE(p_t) = -log_2(p_t) \tag{3.11}$$

Uma interpretação para esta equação é que a entropia associada a cada uma das duas classes diferentes é ponderada pela probabilidade desta classe.

Considerando isso, na função custo focal, adiciona-se um termo $(1 - p_t)^\gamma$ à função custo de entropia cruzada binária. Assim, é apresentada a equação 3.12 a seguir:

$$FL(p_t) = -(1 - p_t)^\gamma log(p_t) \tag{3.12}$$

Observa-se que, se o valor γ for nulo, o custo focal é a entropia cruzada binária. Mas aumentando o valor de γ o coeficiente $(1 - p_t)^\gamma$ fornecerá um peso menor para a entropia cruzada, ou seja, dará uma importância menor, permitindo ponderar a contribuição de diferentes classes.

A contribuição será menor para a classe que está próxima da classificação correta, e por este motivo consegue priorizar as classes de objeto ao invés de classe de fundo. Observa-se ainda que modificando a função custo, o gradiente também é modificado, priorizando sua ação de ajuste nas classificações que estejam erradas.

3.7 Segmentação Semântica

O objetivo da segmentação semântica é associar pixels a classes (fazer a classificação de pixels), o que também permite que sejam classificadas regiões maiores pelo agrupamento dos pixels. Neste sentido, os modelos usados devem fornecer rótulos para os pixels.

Nas primeiras soluções de redes neurais, a rotulação de pixels individuais exigia manter a resolução da imagem, assim, eram usados conjuntos de camadas que não faziam a abstração para um mapa de características, sendo utilizadas camadas totalmente conectadas seguidas de uma operação softmax para fazer a classificação de cada pixel, pela probabilidade de pertencer a classe. O problema é que era obtido grande número de parâmetros, além de necessitar de muito tempo, por serem operações de custo elevado [18]. Neste sentido, uma solução encontrada foi utilizar uma rede neural no modelo de "codificador e decodificador". [29]

Codificadores (enconders) são redes que constroem uma representação interna das características comprimindo informações, processo chamado sub-amostragem e realizado através de redução de dimensionalidade por camadas convolucionais. Assim, os codificadores retornam uma representação resumida das carcterísticas, baseada na entrada. A representação obtida em um mapa de características resumido permite realizar captura de informação de contexto, além disso, permite reduzir o número de parâmetros ajustáveis. Por outro lado, os decodificadores (decoders) realizam a expansão, ou super-amostragem, da imagem, resultando em informações locais, ou seja, retornando a um conjunto com informações de localização, inerente aos pixels [30, 31].

Portanto, o conteúdo da imagem é comprimido em um conjunto de características resumidas e posteriormente é expandido, resultando em uma imagem de resolução maior (igual a original), no entanto, simplificada, ou seja, segmentada. Ambas as etapas são importantes, a primeira permite obter a informações de contexto, pois as características em uma classe são relativamente constantes e podem ser representadas em baixa resolução, o que é útil no processo de classificação que requer o conhecimento da vizinhança de um pixel. Por outro lado, a segunda etapa é importante por recuperar as informações de localização e das fronteiras entre regiões, isso é feito pela extrapolação das classes obtidas na primeira etapa para o conjunto de pixels de alta resolução. [30, 32]

3.7.1 U-Nets

Tendo apresentado uma visão geral do modelo de codificadores e decodificadores, resta observar algumas particularidades.

A primeira abordagem utilizando este modelo foi chamada "rede totalmente convolucional"(FCN, Fully Convolutional Network), substituindo a rede MLP da saída de uma arquitetura para classificação, como a arquitetura LeNet, por uma camada convolucional que realiza a decodificação. No entanto, nesta arquitetura era utilizada uma única camada FCN para a decodificação da imagem. Assim, foi proposto, em seguida, a U-Net [30], que utiliza um conjunto de muitas camadas convolucionais para cada nível, de maneira bastante semelhante à estrutura de codificação, e isso permitiu fazer a propagação de informações de contexto

Além disso, na U-net também são passadas, para as camadas de expansão, informações das camadas de contração com mesma resolução, ou seja, de um mesmo nível. Estas informações contribuem com as informações de localização e de fronteira entre regiões.

Deve-se observar que a U-Net, como foi proposta, visava explorar conjuntos de dados anotados de forma mais eficiente, através de aumentação de dados. Neste sentido, a compressão serve ainda como método de aumentação pois produz um conjunto de sub-imagens maior que a quantidade de imagens originais. No entanto, para que estas sub-imagens não sejam redundantes, as camadas convolucionais são projetadas buscando aproveitar apenas pixels que contenham informação de contexto [30].

3.7.2 Função Custo - Distância de Jaccard

Assim como na detecção de objetos, em problemas de segmentação o desbalanceamento entre classes também é um problema. Neste sentido, uma das funções custo geralmente usadas (dada a natureza do problema onde estão presentes 2 classes, objetos de interesse e fundo da imagem), a Entropia Cruzada Binária (BCE), torna-se insuficiente, visto que a acurácia resultante está relacionada em maior proporção com o fundo, e não com os objetos de interesse, sendo necessário o uso de funções custo que consigam lidar com o desbalanceamento [23].

Como na detecção de objetos, uma métrica de desempenho também usada na segmentação semântica é o índice de Jaccard. Diferente da BCE o índice de Jaccard tem um relacionamento direto com os objetos de interesse, medindo a sobreposição (intersecção sobre união) entre saídas preditas pelo modelo e as anotações de referência.

Como é desejado maximizar o índice de Jaccard, visando obter máxima sobreposição entre estes conjuntos. Para que esta métrica possa ser minimizada pelo algoritmo, a Função Custo Distância de Jaccard, é definida, como é mostrado em [33] pela equação 3.13 a seguir:

$$L_{jaccard} = 1 - JI \tag{3.13}$$

Onde JI é o índice de Jaccard, como foi definido pela equação 3.9

3.8 Descrição de ferramentas utilizadas

3.8.1 Google Street View

O Google Street View (GSV) é uma ferramenta oferecida pelos sistemas da Google (Google Maps e do Google Earth), que permite visualização panorâmica de regiões, presentes em uma base de dados de imagens que foram obtidas ao longo do mundo.

Por outro lado, a GSV API, é um serviço que permite utilizar o GSV de forma automática para a captura imagens. A GSV API utiliza requisições com o protocolo HTTP, para uma identificação de um recurso com um conjunto de parâmetros, definida por uma URL(Uniform Resource Locator). Os principais parâmetros usados na chamada são a localização por coordenadas geográficas (latitude e longitute) do ponto de interesse e data (mês e ano). No entanto, existe ainda um conjunto adicional de parâmetros, consistindo de tamanho da imagem (em pixels), campo de visão (zoom), bússola (ângulo formado com o eixo horizontal da câmera), pitch (ângulo formado com o eixo vertical da câmera), e, finalmente identificação e chave da API.

A respeito da chave e restrições de uso, deve-se observar que o uso da API não é gratuito, embora seja oferecido um crédito inicial para uso. Assim, podem haver limitações para projetos que tenham interesse por utilizá-la.

3.8.2 Sistema Global de Informação sobre Biodiversidade

A *Global Information System Facility* (GBIF) é uma rede internacional, mas também uma infraestrutura, com o objetivo de fornecer acesso aberto a dados dos diversos tipos de seres vivos existentes em todas as regiões do mundo. Assim reune milhões de registos de ocorrência de espécies obtidos através de diversos orgãos, instituições ou indivíduos, que coletam, organizam e disponibilizam este tipo de informação. Para que seja possível reunir estes diferentes colaboradores, o GBIF utiliza um formato de dados padronizado, chamado Darwin Core.

O formato *Darwin Core Standard* (DwC) é um padrão de dados que permite compilar os diferentes tipos dados de biodiversidade, provenientes de fontes variadas e variáveis, de uma maneira simples e eficiente para que possam ser utilizados. Estas características permitem que seja o padrão de dados usado na grande maioria dos registos de ocorrência de espécies, disponíveis no GBIF. Em termos práticos, o padrão DwC está associado ao uso do formato "Darwin Core Archive"(DwC-A) que é um arquivo compactado (no formato ZIP) contendo arquivos de texto com referências e metadados para o acesso dos registros e informações das espécies. Algumas das informações que podem ser obtidas são dados sobre ocorrência (por região, data, tamanho de amotra, etc), mapas e diferentes tipos de mídia (como imagem e áudio) das espécies presentes.

3.9 Sistema Elétrico e Planejamento de Podas

O sistema elétrico consiste no conjunto de equipamentos e instalações utilizados nas tarefas de geração, transmissão e distribuição de energia elétrica.

Dando especial atenção às tarefas de transmissão e distribuição, a transmissão corresponde à interligação entre usinas, responsáveis pela geração de energia, e as regiões

de consumo, no Brasil correspondendo a distâncias de milhares de quilômetros. Por outro lado, a etapa de distribuição, que ocorre após a transmissão, consiste no transporte em distâncias menores, a nível local (ou seja, dentro de um estado ou cidade). [5, 34]

O Sistema de distribuição é dividido em três partes: subtransmissão, rede primária e rede secundária. A subtransmissão é responsável por conectar diferentes cidades e ocorre com níveis de tensão entre 69kV-138kV. Por outro lado, as redes primária e secundária, encontradas dentro das cidades, consistem de linhas de média tensão (entre 2,3kV e 44kV) e baixa tensão (entre 110V e 440V).[34]

As redes de energia elétrica são constituidas de diferentes componentes como linhas (condutores), isoladores, elementos para controle e transformadores. No sistema de transmissão, as linhas são suspensas por torres, que garantem distanciamento, e portanto isolamento entre as linhas. Além disso, a transmissão apresenta subestações que realizam o controle e transformação de níveis de tensão. No sistema de distribuição as linhas são suspensas por postes. Na transmissão também estão presentes subestações, sendo que constituem a interface entre transmissão e subtransmissão, fazendo a transformação dos níveis de tensão entre estes sistemas. Os circuitos condutores originados das subestações de distribuição são linhas de média tensão são chamados de alimentadores. Uma caracterização mais detalhada sobre elementos da rede de distribuição será apresentada a seguir.[5, 35]

3.9.1 Redes de Distribuição de Energia

No sistema de distribuição temos a divisão da rede em primária e secundária. A rede primária é a responsável pela distribuição ao longo das cidades e entrega de energia a consumidores primários ou a transformadores de distribuição, utilizados na transformação entre os níveis de tensão primário e secundário (ou seja, de média tensão para baixa tensão), enquanto a rede secundária é responsável pela entrega da energia a consumidores secundários. É possível fazer ainda a divisão da rede, de forma simplificada, em 4 tipos (aplicados para redes primárias e secundárias)[36]: rede de distribuição aérea convencional (RDA), rede de distribuição aérea compacta (RDP), rede de distribuição aérea isolada (RDI) e rede de distribuição subterrânea (RDS).

A rede aérea convencional (RDA) é caracterizada por condutores de alumínio sem cobertura protetora (ou seja, condutores nus). Neste sentido, é o tipo mais vulnerável de rede, onde os condutores ficam expostos. Em relação à rede convencional primária, os condutores ficam suspensos em postes dispostos horizontalmente sobre isoladores e elementos para sustentação mecânica chamados cruzetas. Em relação à rede convencional secundária, os condutores estão dispostos em posição vertical, com fixação através de elementos de sustentação chamados braços (tipo C e L), espaçadores e outros elementos que complementam a sustentação, como estribos (alças metálicas) e conectores. [37, 38]

No caso da rede compacta (RDP), se diferencia da rede convencional no que se refere ao espaçamento entre condutores, que é menor, assim as estruturas utilizadas são mais "compactas". Para que seja possível utilizar espaçamentos menores, a RDP primária utiliza condutores envoltos por uma cobertura protetora, ou seja, condutores protegidos. Estes não oferecem total isolamento, portanto os cabos não devem entrar em contato pois podem entrar em curto, mas são permitidos eventuais contatos dos condutores energizados com galhos e folhas de árvores. Paralelamente, na RDP secundária também podem ser utilizados espaçadores, no entanto, mais recentemente também tem sido utilizado o padrão de cabos multiplexados, onde os condutores (com exceção do neutro) são totalmente isolados podendo, portanto, entrar em contato, neste caso formando uma trança, este tipo de rede é a chamada rede isolada (RDI). [39, 40, 36]

Redes compactas são adequadas para locais com arborização, visto que, a compactação proporcionada, pelos cabos protegidos reduz a área de podas necessária. A compactação também é adequada para situações com congestionamento de circuitos. Redes isoladas também são adequadas para situações com arborização e congestionamento de circuitos, além de oferecerem segurança contra choques elétricos. Outra vantagem de cabos protegidos e isolados é que dificultam a ocorrência de furtos. Existem outras situações de interesse que podem ser encontradas em um estudo mais aprofundado, em todo caso, pode-se dizer que ambas são mais robustas e confiáveis que as redes convencionais.[39, 36, 40]

Finalmente, na rede subterrânea (RDS) os cabos também são isolados, no entanto, estão localizados em dutos subterrâneos, o que evita o conflito com copas de árvore, e embora ainda possam haver conflitos com raizes, grande parte dos problemas de interação com vegetação presentes em redes aéreas são resolvidos.[41]

No Brasil a rede aérea ainda é a mais comum, o que pode ser explicado pelo menor custo de instalação e manutenção. Neste sentido, a necessidade de realização de podas continuará presente principalmente considerando a extensão por milhares de quilômetros das redes de transmissão e distribuição, para a qual haveria grande custo de substituição.

3.9.2 Planejamento de Podas

O Brasil é o país com a maior biodiversidade do mundo. No entanto, é um país urbano, onde mais de 80% da população reside em cidades, portanto o manuseio da biodiversidade, em especial da vegetação, em áreas urbanas é de grande importância, podendo ser enumeradas diversas contribuições positivas geradas pela arborização urbana como redução da poluição do ar, diminuição de ilhas de calor formadas pela impermebilização e ocupação acentuada do espaço urbano, interceptação das águas da chuva e contribuições para ciclo da água, redução de ruído, além de proporcionar a manutenção da biodiversidade para espécies animais que possam ser abrigadas pela vegetação.

Neste contexto, o contato de árvores com a rede de distribuição é uma fonte de diferentes problemas, podendo levar a interrupções no fornecimento, acidentes e danos a equipamentos. De modo geral, este contato ocorre pelo crescimento desordenado de algumas espécies ou pela queda de árvores e galhos que ocorre principalmente em condições de chuvas e ventanias. No entanto, existe um conjunto de situações diversas que precisam ser analisadas, seja pela ocasião em que a interação poderá ocorrer, por exemplo, se já ocorre, ocorrerá no futuro ou até mesmo se poderá ocorrer caso uma instalação seja iniciada. Considerando as diferentes espécies, deve-se levar em conta fatores como o tamanho da copa, a taxa de crescimento da espécie e, além disso o tipo de instalação presente em proximidade àquela espécie. A norma NBR 16246-3 permite estabelecer critérios para avaliação do risco proporcionado por árvores através de diferentes níveis de análise. Para o conjunto de situações possíveis, uma medida para o manejo da arborização, visando a compatibilização entre arborização e componentes urbanos, em especial a rede de distribuição, é a poda, consistindo no ato de cortar ramos vivos ou mortos de árvores, mantendo-se sua fitossanidade.

3.9.2.1 Tipos de Podas

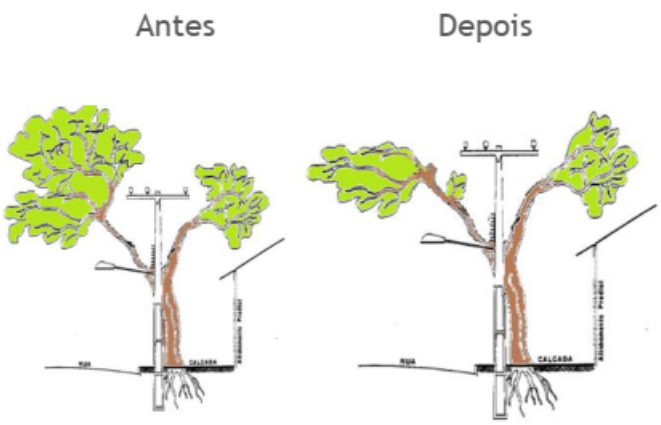
O conjunto de situações considerado é atestado pela norma NBR 16246-1 que recomenda a aplicação de poda ou remoção de árvores em casos em que árvores ou galhos estejam crescendo abaixo ou dentro da área de passagem da rede elétrica, podendo ser feita pela remoção de galhos inteiros ou remoção dos galhos que tenham ramos laterais crescendo em direção à faixa de segurança. Para o detalhamento destas ações, podem ser considerados diferentes tipos de poda.

Em situações em que a copa das árvores ainda não alcançou a rede elétrica, especialmente em áreas de implantação de arborização, são aplicadas podas de formação, que ainda na fase de desenvolvimento da planta, direcionam seu crescimento para que não cresçam em direções que possam levar ao contato com elementos da rede de distribuição. A Figura 6 ilustra uma poda de formação.

Em árvores adultas, para manter um controle do comprimento das ramificações sem alteração do formato original da copa, são aplicadas podas de rebaixamento (também chamadas podas de contenção), as quais poderão ser utilizadas para que as ramifições não alcancem a rede elétrica e seja mantida a vitalidade da árvore. Neste caso, deve-se tomar cuidado para que não sejam realizadas podas drásticas, isso porque além de prejudicarem a vitalidade das árvores resultam em novas ramificações com crescimento rápido, assim podendo causar novos problemas. A figura 7 representa a finalidade de poda de rebaixamento.

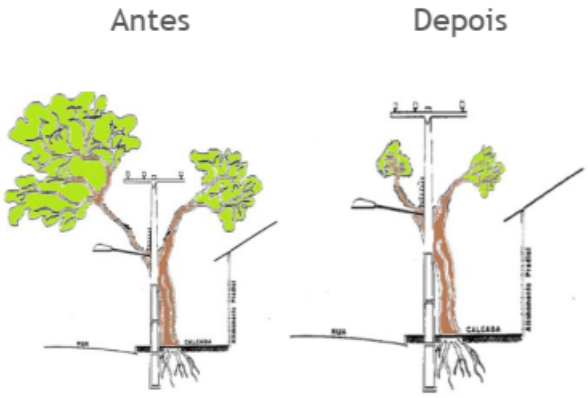
Outro tipo de poda é a poda de segurança (ou reparação) que tem como objetivo livrar a rede de distribuição em situações críticas e também visando corrigir problemas da

Figura 6 – Poda de Formação



Fonte: Referência [41]

Figura 7 – Poda de Rebaixamento



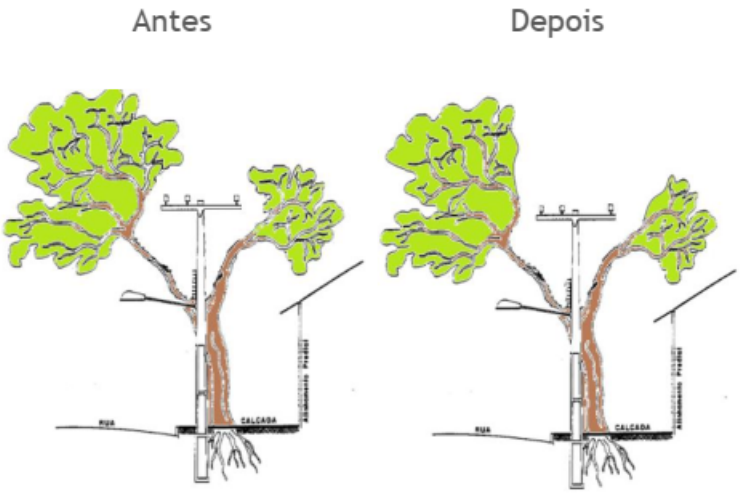
Fonte: Referência [41]

aplicação incorreta da poda de formação. Neste caso a poda é efetuada pela remoção de ramos e tomando o cuidado para que os galhos demorem para alcançar a parte superior. A poda pode ser realizada pelas técnicas em "V"ou em "furo". A figura 8 representa a finalidade de poda de segurança.

3.9.2.2 Critérios para realização de podas

Ainda considerando a avaliação de risco proporcionado por árvores para a tomada de decisão na aplicação de podas, temos ainda como atributo o tipo de rede que poderá ser afetado pelo contato das árvores. Assim, considerando os tipos de rede apresentados em 3.9.1, é apresentada a Tabela 3 relacionando valores de distâncias de segurança mínima que devem ser encontradas após a poda de acordo com o tipo de rede presente.

Figura 8 – Poda de Segurança



Fonte: Referência [41]

Tabela 3 – Distância de Segurança mínima após a poda de acordo com tipo de rede [4]

Tipo de rede	Distância de segurança mínima medida após a poda
Rede de alta tensão em 138 kv	4,30 m
Rede de alta tensão em 69 kv	4,00 m
Rede convencional ou protegida de média tensão em 34,5 kv	2,00 m
Rede convencional ou protegida de média tensão em 13,8 kv	2,00 m
Rede convencional de baixa tensão em 110 ou 220 kv	1,00 m

4 Metodologia

Dentre as tarefas de interesse deste trabalho, necessárias para a definição de estratégias de podas, temos a identificação de elementos da infraestrutura de distribuição de energia, como postes e condutores. Neste sentido, em [3] é considerado que fatores como a diversidade de componentes da rede em especial, diferentes tipos de postes, e além disso a similaridade de alguns tipos de árvores com postes, em aspectos de textura, são possíveis causas para piores desempenhos na classificação de imagens através de métodos baseados em características de forma. Assim, faz sentido que sejam utilizadas estratégias de DL neste contexto. O trabalho [7] também considera esta uma abordagem promissora, em vista da simplificação proporcionada pelos métodos.

Baseando-se principalmente nos trabalhos [1], [2] e [9], foi utilizada a API REST do GSV para obtenção de imagens de postes e condutores, assim como imagens de árvores. Estas imagens podem ser utilizadas na definição de critérios de poda, assim como da periculosidade dos cenários. Portanto, foram utilizadas as coordenadas dos postes, fornecidas por uma concessionária de distribuição de energia e disponibilizadas pela empresa "Daimon Engenharia e Sistemas".

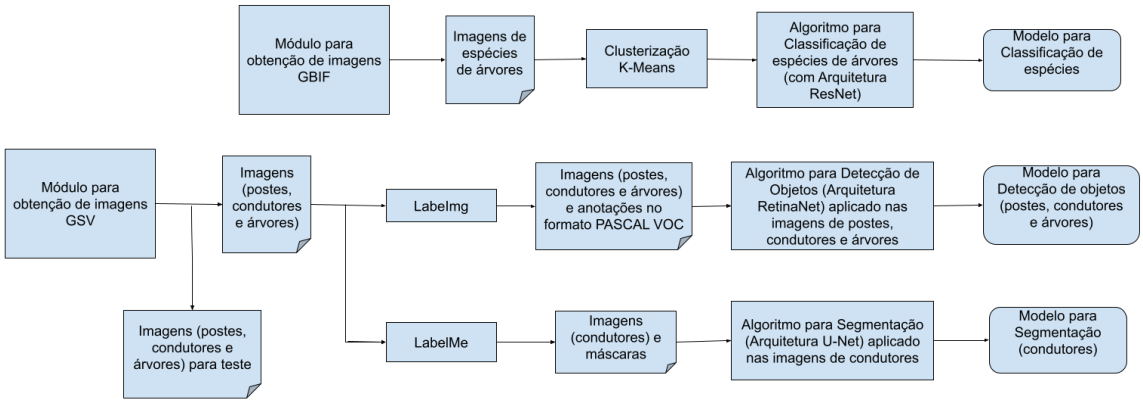
Por outro lado, na definição de estratégias de podas, além da detecção de elementos da infraestrutura de distribuição de energia, também é necessário que seja realizada a classificação de espécies de árvores presentes nas proximidades destes elementos. Deste modo, além das imagens obtidas pelo GSV, também foi utilizado o "Sistema Global de Informação sobre Biodiversidade"(GBIF) permitindo obter imagens de diferentes espécies de árvores.

Foi proposto o seguinte fluxo de processamento:

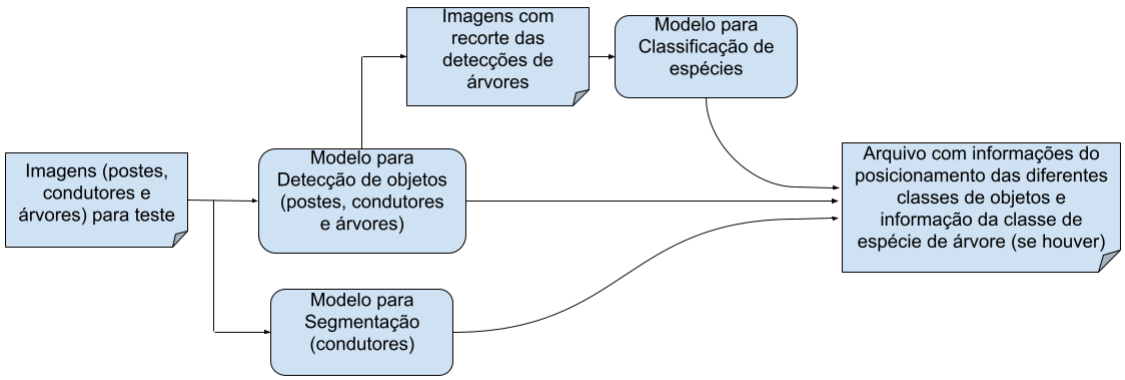
- Utilizar coordenadas fornecidas pela concessionária de distribuição de energia, para obter imagens dos elementos de infraestrutura, em imagens panorâmicas do GSV, com ângulos de vista e níveis de resolução diferentes.
- Realizar o treinamento e avaliação de desempenho da arquitetura RetinaNet na tarefa de detecção de objetos considerando elementos de infraestrutura e árvores.
- Verificar a existência de datasets e catálogos com imagens de espécies de árvores da região geográfica de interesse no GBIF, aplicando-os ao treinamento de algoritmos para classificação de espécies nas imagens do GSV onde houve detecção de árvores.

O conjunto de procedimentos é apresentado nos diagramas da Figura 9 a seguir:

Figura 9 – Diagrama da Metodologia



(a) Obtenção dos Modelos



(b) Aplicação dos Modelos

4.1 Procedimentos para Obtenção de Datasets

Na obtenção das imagens de postes foram utilizados como parâmetros, além das coordenadas geográficas dos postes, resolução de imagem = 640x600, campo de visão ("zoom") = 30º, 50º, 60º e 90º, ângulo de orientação no eixo vertical = 0º e ângulo de orientação no eixo horizontal = 0º, 90º, 180º e 270º.

Como justificativa para a escolha dos parâmetros, primeiramente a resolução é a resolução máxima possível de ser obtida pelo GSV para o conjunto de cordenadas analisado. Em segundo lugar, a escolha por diferentes valores para campo de visão foi dada buscando aumentar a diversidade de vistas. Isso porque para determinadas vistas um campo de visão menor poderia impossibilitar a visualização dos objetos de interesse, enquanto que um campo de visão maior levaria a perda na resolução. Analogamente, a escolha por diversos

ângulos de orientação horizontal também foi feita buscando aumentar a diversidade de vistas, com maior chance de visualização dos objetos em cada coordenada.

No caso das imagens dos condutores e também de árvores, a obtenção foi feita através das mesmas coordenadas, ou seja, coordenadas de postes, por serem as únicas coordenadas disponíveis. Nestes casos apenas foi dado maior destaque para os objetos de interesse, condutores ou árvores.

Neste sentido, para a obtenção das imagens de condutores foi dada preferência por imagens em vista frontal, ou seja, utilizando ângulo de orientação no eixo horizontal = 0^o e 180^o. Além disso, resolução de imagem = 640x600, campo de visão ("zoom") = 30^o, 50^o, 70^o e 90^o, e ângulo de orientação no eixo vertical = 0^o.

Como justificativa para a escolha dos parâmetros deste caso, a resolução é a resolução máxima possível de ser obtida pelo GSV. A escolha por diferentes valores para campo de visão e ângulos de orientação horizontal foi dada buscando aumentar a diversidade de vistas com maior chance de visualização dos objetos em cada coordenada. Neste caso, entretanto, restringindo-se a 0^o e 180^o para ângulos de orientação horizontal. Isso porque no caso de condutores, no conjunto de coordenadas analisado, foi observada maior qualidade (considerando constraste e luminosidade), para visualização dos condutores, nas imagens com vista frontal.

Para a obtenção das imagens de árvores pelo GSV (ainda direcionadas à tarefa de detecção de objetos), foi dada preferência por imagens em vista ortogonal, ou seja, utilizando ângulo de orientação no eixo horizontal = 70^o e 250^o. Além disso, resolução de imagem = 640x600, campo de visão ("zoom") = 70^o, 90^o e 100^o, e ângulo de orientação no eixo vertical = 0^o.

Como justificativa para a escolha dos parâmetros deste caso, a resolução é a resolução máxima possível de ser obtida pelo GSV. A escolha por diferentes valores para campo de visão e ângulos de orientação horizontal foi dada buscando aumentar a diversidade de vistas com maior chance de visualização dos objetos em cada coordenada. Neste caso, entretanto, a escolha por ângulos de orientação no eixo horizontal = 70^o e 250^o foi feita buscando obter uma separação maior entre árvores individuais e o fundo das imagens, que no caso de vista frontal poderia conter outras espécies, dificultando a tarefa de rotulação, devido a sobreposição entre regiões semelhantes (árvore de interesse e fundo com outras árvores).

Após a obtenção dos diferentes conjuntos de imagens, para a preparação do dataset para a tarefa de detecção de objetos é feita a rotulação das imagens definindo caixas delimitadoras envolvedo os objetos de interesse. A rotulação é realizada no software LabelImg sendo gerados arquivos com as classes e anotações que serão aplicados como referência (ground-truth) ao algoritmo, junto às imagens, no formato "Pascal VOC".

Um aspecto a ser observado é que as anotações para imagens contendo árvores foi feita com a separação entre 3 tipos de árvores diferentes. "Tipo 1"com copa maior e mais densa, "Tipo 2"com copa menor e menos densa, "Tipo 3"com copa de folhas esparsas e "Tipo 4"com copa com a morfologia característica das palmeiras (família Arecaceae) [42].

4.2 Parâmetros para Detecção de Objetos

O modelo utilizado na tarefa de detecção de objetos, é arquitetura para detecção de objetos RetinaNet. Isso por ainda estar entre os modelos com melhor desempenho, conforme apresentado em [43], e além disso, devido à grande popularidade apresentada, possuindo boas referências para a implementação.

A implementação utilizada, é uma implementação no framework Keras disponibilizada em [44]. A implementação se baseia diretamente na descrição fornecida por [25] que utiliza um backbone FPN sobreposto a um backbone ResNet, e contém 2 submodelos para as tarefas de classificação e regressão, que são aplicados às saídas do backbone. Além disso, permite o uso de formatos de dados como "Pascal VOC", "MS COCO"e "CSV"como entradas para o algoritmo de treino.

No treinamento do modelo (RetinaNet), no caso de postes foi utilizado um dataset com 1650 imagens, dentre estas, sendo definidas 365 imagens com anotações de postes. No caso de condutores, foi utilizado um dataset com 400 imagens, dentre estas, sendo definidas 125 imagens com anotações de condutores. No caso de árvores, foi utilizado um dataset com 605 imagens, dentre estas, sendo definidas 425 imagens com anotações de árvores. Todos datasets foram obtidas com os critérios definidos para obtenção de imagens no GSV, considerados acima. A diferença na quantidade de imagens utilizada para os diferentes objetos se deve simplesmente pela escassez de recursos humanos. Isso porque a rotulação das imagens é uma tarefa trabalhosa e que demanda grande quantidade de tempo, existindo, inclusive, empresas que oferecem serviços para rotulação de dados para tarefas de aprendizado de máquina. No entanto, tais serviços não foram utilizados neste trabalho.

Além do formato do conjunto de dados utilizado, na implementação de [44] são utilizados como parâmetros do treinamento:

- Aumentação de dados utilizando intervalos para as transformações de: rotação, translação, escala, cisalhamento, giro, contraste, luminosidade, hue e saturação.
- Tamanho de Batch, taxa de aprendizado e quantidade de épocas do treino.
- Otimização da função de custo é feita com o algoritmo Adam.

Os parâmetros considerados podem ser ajustados na chamada do método de treino do modelo. No caso da aumentação de dados, apenas pode ser feita a escolha por aplicá-la ou não, sendo que para que sejam alterados os intervalos das transformações deve ser modificada a codificação interna da implementação.

O segundo conjunto de parâmetros associados estão mais relacionados com o modelo proposto. São apresentados como:

- Reutilização de pesos por transferência de aprendizado, utilizando como modelo base a arquitetura Resnet50 pré-treinada com o dataset COCO, e com congelamento das camadas do modelo base
- Tamanho de kernel e Tamanho de stride usados na definição dos submodelos
- Quantidade de filtros das camadas convolucionais iguais a 256 para classificação e regressão.
- Inicialização das camadas convolucionais com $bias = 0$ e pesos aleatórios com distribuição gaussiana com $\sigma = 0.01$ para todas as camadas com exceção da ultima camada de classificação que é definida de acordo com índice de confiança $\pi = 0.01$
- Conjuntos de valores para razões de aspecto e escala das âncoras para cada posição
- Número de classes para o submodelo de classificação = 1.

Dentre os parâmetros considerados, podem ser ajustados na chamada do método de treino os conjuntos de valores para razão de aspecto e escala das âncoras através de um arquivo de configuração.

A respeito do backbone FPN, a construção das camadas é feita exatamente da mesma forma (ou seja, com as mesmas operações) proposta em [25], onde as camadas piramidais de níveis P3 a P7 são construídas a partir das camadas C3 a C5 da ResNet, sendo utilizado em todos os níveis camadas com um conjunto de 256 filtros.

Os parâmetros da construção do modelo, como tamanho de kernel, e tamanho de stride são definidos através do arquivo de configuração. É recomendado que alterações não sejam realizadas nestes parâmetros. Os parâmetros da construção dos submodelos como quantidade de filtros das camadas convolucionais e valores de inicialização de pesos e bias das camadas convolucionais também são definidos de acordo com [25].

Temos também o número de classes que depende da quantidade de classes do problema de interesse e o número de valores para o submodelo de regressão igual a 4, correspondendo aos valores xmin, xmax, ymin, ymax para as caixas delimitadoras.

É relevante considerar, que conforme é apresentado em [25] parâmetros associados com decisões de mais alto nível na definição do modelo são mais importantes que

modificações nos hiperparâmetros. Deste modo, as caracteríticas das âncoras e o uso do backbone FPN são de grande importância por estarem relacionadas à densidade de âncoras, e também ao aproveitamento de características multi-escala.

Por fim, temos ainda parâmetros associados com a avaliação do modelo. O modelo é treinado visando minimizar os valores de custo para classificação e regressão, e utilizando como métrica de interesse o valor mAP.

Para a função de custo Focal, usada na tarefa de classificação, os parâmetros definidos na implementação de [44] são $\alpha = 0.25$ e $\gamma = 2$.

Para a função de custo Smooth L1, usada na tarefa de regressão das caixas delimitadoras, a implementação é baseada diretamente na definição fornecida por [26] onde a função de custo Smooth L1 corresponde a uma combinação das regressões Lasso (L1) com fator linear e Ridge (L2) com fator quadrático sobre a diferença $x = t - v$, que é a diferença entre as coordenadas das predições para as caixas delimitadoras (t) e as refererências (v). É aplicado um peso $\sigma = 3$ internamente à função Smooth L1. O parâmetro σ corresponde ao balanço entre as duas funções de custo de interesse.

Em ambos os submodelos a otimização da função de custo é feita com o algoritmo de otimização Adam.

4.3 Parâmetros para Segmentação Semântica

Embora os resultados obtidos utilizando detecção de objetos através da arquitetura RetinaNet tenham possibilitado a determinação dos critérios de podas para as situações presentes através da localização (determinação das coordenadas) dos objetos de interesse, no caso da detecção de condutores foram observadas limitações na abordagem. Isso porque no primeiro caso temos elementos visualmente concentrados e que se encontram alinhados com os eixos das caixas delimitadoras geradas pela arquitetura RetinaNet, enquanto que, no caso dos condutores isso não acontece, pois são elementos que podem estar distribuídos por toda a imagem e com forma rotacionada, de modo que a localização se torna imprecisa. Assim optou-se por utilizar um método de segmentação através da arquitetura U-Net, a qual realiza o processo de segmentação semântica.

Conforme apresentado por [45] a U-Net ainda se encontra entre as melhores propostas para a tarefa de segmentação, contando com boas referências para a implementação.

Por se tratar da tarefa de segmentação, é utilizado um conjunto de máscaras e imagens. As máscaras são geradas por anotações produzidas no software Labelme.

No treinamento do modelo, foram utilizadas 405 imagens de condutores, cada uma com uma máscara associada. As imagens foram obtidas seguindo os mesmos critérios para obtenção de imagens no GSV apresentados nas tarefas anteriores. Portanto, foram

utilizados como parâmetros, além das coordenadas geográficas dos postes, resolução de imagem = 640x600, campo de visão ("zoom") = 30º, 50º, 70º, ângulo de orientação no eixo vertical = 0º e ângulo de orientação no eixo horizontal = 0º, 180º. Os

Da mesma forma que as imagens de condutores na detecção de objetos, foi feita a restrição dos ângulos de orientação horizontal para 0º e 180º buscando destacar as melhores vistas para esta classe de objetos.

A implementação do modelo foi feita através do framework Keras, baseando-se na descrição de [30] para o modelo U-Net. Portanto é definido um caminho de contração (codificador) seguido por um caminho de expansão (decodificador). Além disso sendo utilizada a concatenação de camadas de mesmo nível do caminho de contração para o de expansão.

Portanto os parametros do caminho de contração são:

- Camada de entrada com tamanho: (128, 128, 3) correspondendo à Largura, Altura e canais de cores.
- Nas camadas convolucionais: Número de filtros = [16, 32, 64, 128, 256], Tamanho de kernel = 3, Padding mantendo o tamanho da imagem e Função de ativaação ReLU
- Inicialização das camadas convolucionais com distribuição normal truncada (ou seja, com restrição de domínio) com média zero.
- Camada de MaxPooling de tamanho = 2
- Dropout de 30%, 20% e 10%, para os respectivos níveis

Observa-se que para cada nível do caminho de contração é realizado downsampling através das camadas de MaxPooling, e que a cada downsampling a quantidade de filtros é dobrada. Além disso, na camada de entrada as imagens são redimensionadas para que o tamanho seja divisível por 2^n, onde n é número de níveis.

Para o caminho de expansão, os parâmetros são:

- Nas camadas convolucionais: Número de filtros = [256, 128, 64, 32, 16], Tamanho de kernel = 3, Padding mantendo tamanho da imagem e Função de ativaação ReLU
- Inicialização das camadas convolucionais com distribuição normal truncada (ou seja, com restrição de domínio) com média zero.
- Camada de Convolução Transposta em substituição ao MaxPooling, com tamanho = 2

- Concatenção dos níveis Ci de contração (iniciando com nível superior = 1) e Ej de expansão (iniciando com nível inferior = 1): C4-E1, C3-E2, C2-E3, C1-E4.
- Dropout de 10%, 20% e 30%, para os respectivos níveis.
- Camada de saída com tamanho: (128, 128, 1) correspondendo à Largura, Altura e 1 canal de cor.

Como parâmetros do treinamento do modelo temos:

- Tamanho de Batch, taxa de aprendizado e quantidade de épocas do treino.
- Divisão do conjunto de entradas em treino e teste com proporção 80% para treino e 20% para teste.
- Otimização da função de custo é feita com os algoritmo Adam e SGD.

Tais parâmetros podem ser ajustados na chamada do método de treino do modelo.

Por fim, temos ainda parâmetros associados com a avaliação do modelo. O modelo é treinado visando minimizar a função de custo Distância de Jaccard utilizando como métrica de interesse, para avaliação, o coeficiente de Jaccard. A otimização da função custo é feita com os algoritmos Adam e SGD fazendo comparações do desempenho de ambos, a escolha é devida ao seu uso com comprovada eficácia, por um lado em termos de velocidade de convergência, e por outro, em termos de capacidade de generalização, verificada em trabalhos recentes da área [46].

4.4 Parâmetros para Classificação de Espécies de Árvores

Na obtenção das imagens de espécies de árvores foram selecionadas, por simplicidade, três espécies de interesse: “Anacardium Occidentale L.”, “Cocos Nucifera L.”, e “Mangifera Indica L.” As imagens também foram obtidas utilizando a API REST do GBIF. As imagens são obtidas no formato “Darwin Core” conforme disponibilizadas pelo GBIF, sendo possível a extração de formatos comuns de imagem a partir deste.

Tendo as imagens de cada uma das três espécies, antes de introduzi-las no algoritmo para classificação, foi realizado um pré-processamento visando extrair as imagens com maior semelhança com o conjunto de imagens do problema de interesse, de modo que pudessem ser aproveitadas informações dessas imagens neste problema.

A extração foi realizada primeiramente selecionando 300 imagens de cada uma das espécies de maneira aleatória, e em seguida aplicando o algoritmo K-Means a cada um dos conjuntos de imagens, agrupando assim as imagens com características mais próximas. Após

o agrupamento, foram selecionadas, por visualização "a olho nu", os agrupamentos coms imagens mais semelhantes ao conjunto de imagens de interesse, portanto, agrupamentos com imagens de espécies de árvores caracterizadas por vista panorâmica. Observa-se que poderiam ter sido aplicados algortimos para fazer a seleção por medidas de similaridade entre os dois conjuntos (imagens do GBIF e do GSV), no entanto, por simplicidade, e para os fins deste trabalho, decidiu-se pela seleção visual "a olho nu".

Tendo o conjunto de imagens das três espécies, e anotações de suas classes, estas são introduzidas no algoritmo de classificação. No algoritmo são utilizadas as seguintes configurações:

- Divisão do conjunto de entradas em treino e teste com proporção 85% para treino e 15% para teste.
- Aumentação de dados proporcionada pela classe 'ImageDataGenerator' do framework Keras.
- Transferência de aprendizado utilizando a arquitetura ResNet50V2 com pesos pré-treinados do dataset ImageNet, e com congelamento do modelo base.
- Camada de entrada igual a camada de entrada do modelo base ResNet50V2, ou seja, camada com dimensões (128, 128, 3) correspondendo a altura, largura e número de canais.
- Camada de saída modificada para 3 unidades (correspondentes às 3 classes) com função de ativação Softmax.
- Função de custo Entropia Cruzada Categórica, utilizando como métrica de interesse a acurácia.
- Algoritmo de otimização da função de custo, Adam.

4.5 Experimentos para o Ajuste de Parâmetros

Para o entendimento dos modelos e definição dos melhores parâmetros para as tarefas e modelos implementados, foram realizados diferentes testes para as diferentes implementações, que serão apresentados a seguir.

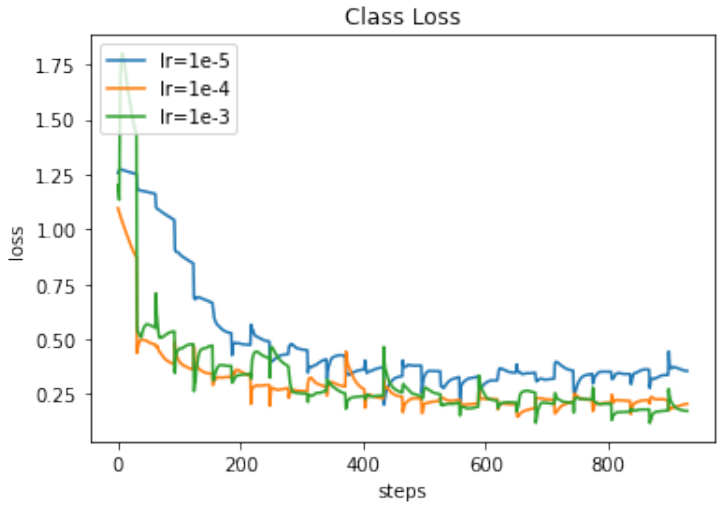
4.5.1 Detecção de Objetos (RetinaNet)

4.5.1.1 Taxa de Aprendizado

A figura 10 a seguir apresenta as curvas de custo para o submodelo de classificação da arquitetura RetinaNet. São utilizadas diferentes taxas de aprendizado para o algoritmo

Adam, ou seja, diferentes inicializações para a taxa, que posteriormente é ajustada dinamicamente conforme a definição do algoritmo Adam. É utilizada a configuração padrão da implementação, com uso de aumentação de dados, com reutilização dos pesos da ResNet50 pré-treinados no ImageNet, e como parâmetros da chamada do método de treino: tamanho de batch = 8, quantidade de épocas = 30.

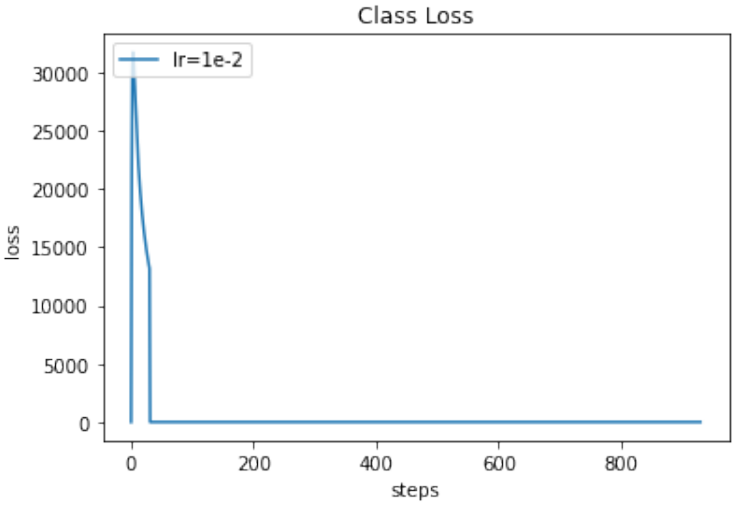
Figura 10 – Custo de Classificação para diferentes taxas de aprendizagem



Fonte: Autor

A figura 11 a seguir utiliza as mesmas configurações da figura 10 para a taxa de aprendizado lr = 1e-2.

Figura 11 – Custo da Classificação para taxa de aprendizagem lr = 1e-2



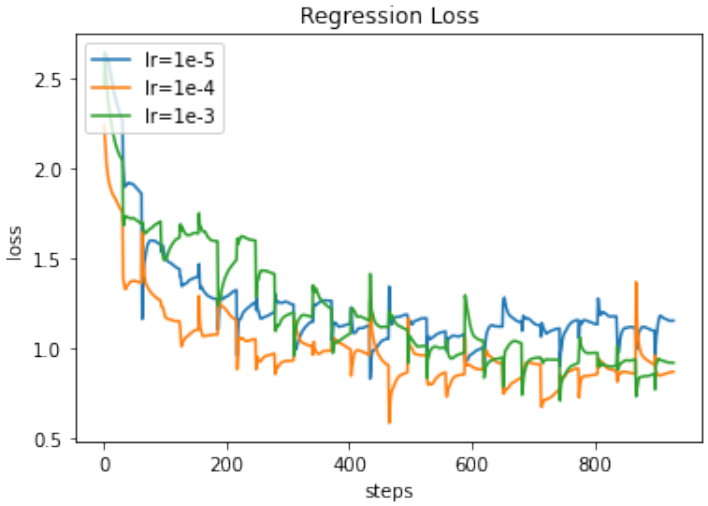
Fonte: Autor

Portanto, pelas figuras 10 e 11 pode-se considerar que a taxa de aprendizado aplicada influência na velocidade de convergência do algoritmo de otimização do submodelo de

classificação, assim como no valor alcançado para o custo. Observa-se ainda que a partir de um determinado valor, lr=1e-2, para a taxa de aprendizado, o algoritmo diverge.

A figura 12 a seguir apresenta as curvas de custo para o submodelo de regressão da arquitetura RetinaNet, para diferentes taxas de aprendizado, utilizando a configuração padrão da implementação, com uso de aumentação de dados, com reutilização dos pesos da ResNet50 pré-treinados no ImageNet, e como parâmetros da chamada do método de treino: tamanho de batch = 8, quantidade de épocas = 30.

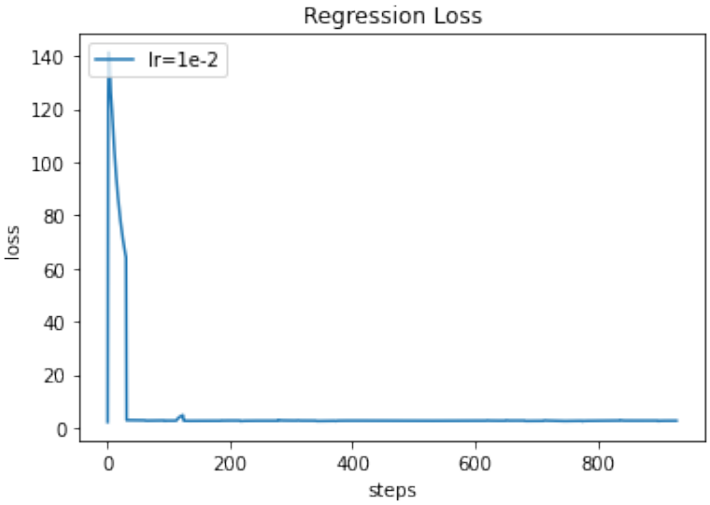
Figura 12 – Custo de Regressão para diferentes taxas de aprendizagem



Fonte: Autor

A figura 13 a seguir utiliza as mesmas configurações da figura 12 para a taxa de aprendizado lr = 1e-2.

Figura 13 – Custo de Regressão para taxa de aprendizagem lr = 1e-2



Fonte: Autor

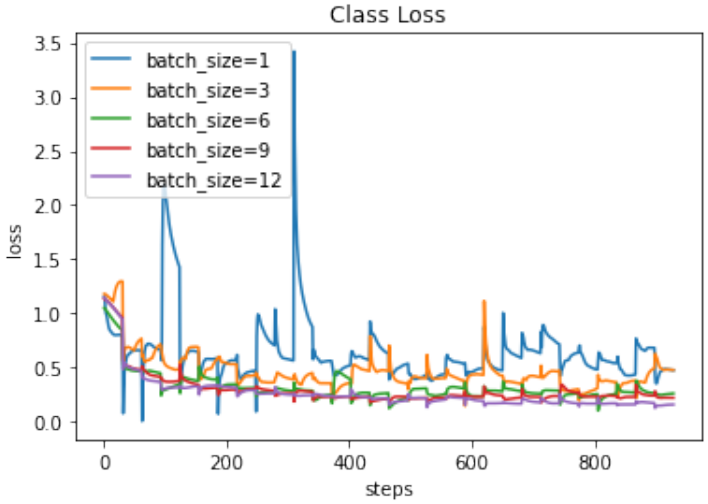
Portanto, pelas figuras 12 e 13 pode-se considerar que a taxa de aprendizado aplicada também influência na velocidade de convergência do algoritmo de otimização do

submodelo de regressão, assim como no valor alcançado para o custo. Observa-se que a partir do valor, lr=1e-2, para a taxa de aprendizado, o algoritmo também diverge.

4.5.1.2 Tamanho de Batch

A figura 14 a seguir apresenta as curvas de custo para o submodelo de classificação da arquitetura RetinaNet, para diferentes tamanhos de batch, utilizando a configuração padrão da implementação, com uso de aumentação de dados, com reutilização dos pesos da ResNet50 pré-treinados no ImageNet, e como parâmetros da chamada do método de treino: taxa de aprendizado = 1e-4, quantidade de épocas = 30.

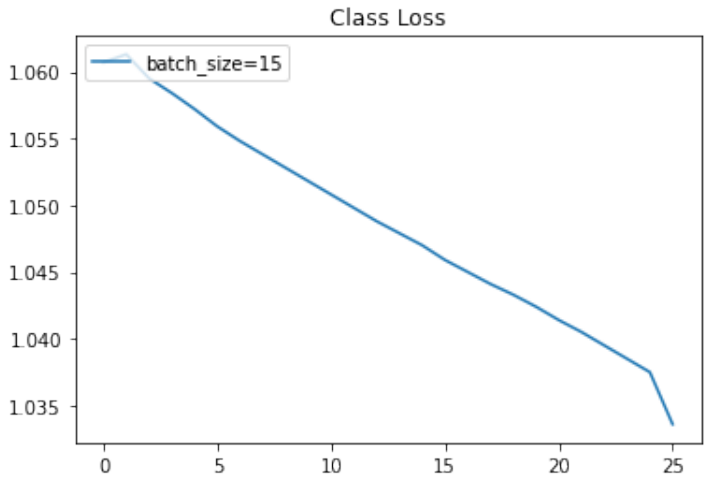
Figura 14 – Custo de Classificação para número de imagens diferentes



Fonte: Autor

A figura 15 a seguir utiliza as mesmas configurações da figura 14 para o tamanho de batch = 15.

Figura 15 – Custo de Classificação para tamanho de batch = 15

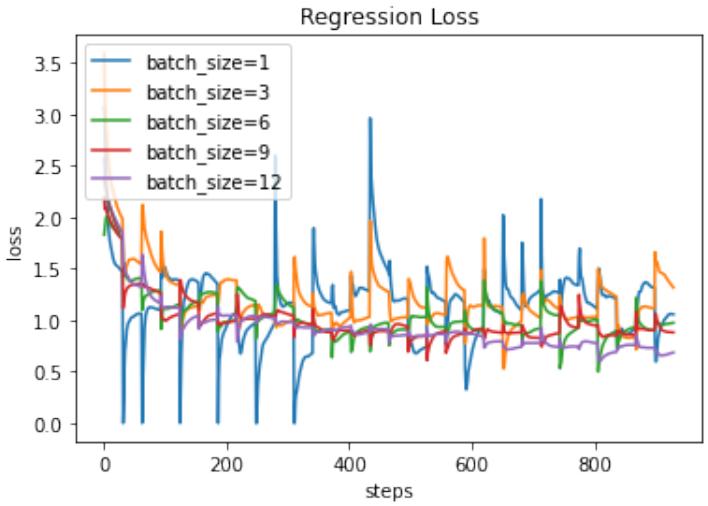


Fonte: Autor

Portanto, pelas figuras 14 e 15 pode-se considerar que alterações no tamanho de batch influenciam a convergência do algoritmo de otimização no submodelo de classificação. Observa-se que para um bs = 12 a curva apresenta oscilações menores. Por outro lado, para um valor de bs = 15 o algoritmo diverge.

A figura 16 a seguir apresenta as curvas de custo para o submodelo de regressão da arquitetura RetinaNet, para diferentes tamanhos de batch, utilizando a configuração padrão da implementação, com uso de aumentação de dados, com carregamento dos pesos da ResNet50 pré-treinados no ImageNet, e como parâmetros da chamada do método de treino: taxa de aprendizado = 1e-4, quantidade de épocas = 30.

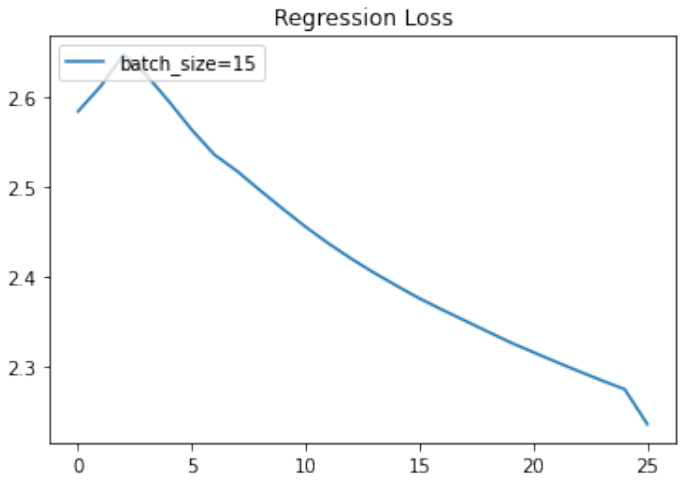
Figura 16 – Custo de Regressão para tamanhos de batch diferentes



Fonte: Autor

A figura 17 a seguir utiliza as mesmas configurações da figura 16 para o bs = 15.

Figura 17 – Custo de Regressão para tamanho de batch = 15



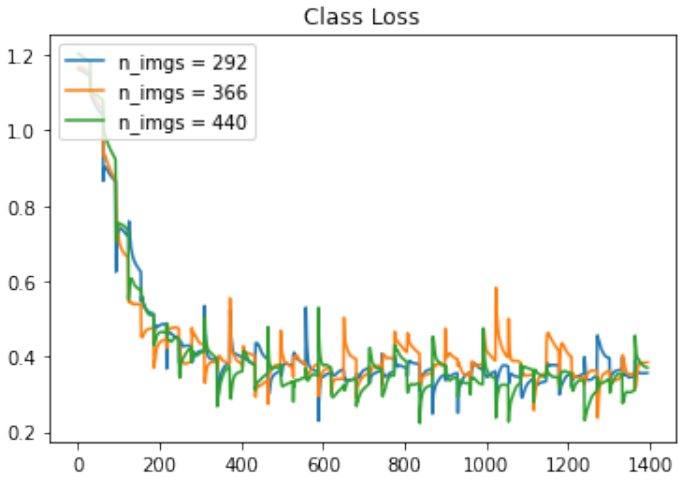
Fonte: Autor

Portanto, pelas figuras 16 e 17 pode-se considerar que alterações no tamanho de batch também influenciam a convergência do algoritmo de otimização no submodelo de regressão. Observa-se que para um bs = 12 a curva apresenta oscilações menores. Por outro lado, para um valor de bs = 15 o algoritmo diverge.

4.5.1.3 Quantidade de Anotações

As figuras 18 e 19 a seguir apresentam as curvas de custo para os submodelos de classificação e regressão da arquitetura RetinaNet, para datasets com quantidade de anotações diferente para o conjunto de imagens. É utilizada a configuração padrão da implementação, uso de aumentação de dados, e como parâmetros da chamada do método de treino: tamanho de batch = 8, taxa de aprendizado = 1e-5, quantidade de épocas = 45.

Figura 18 – Custo de Classificação para número de anotações diferente



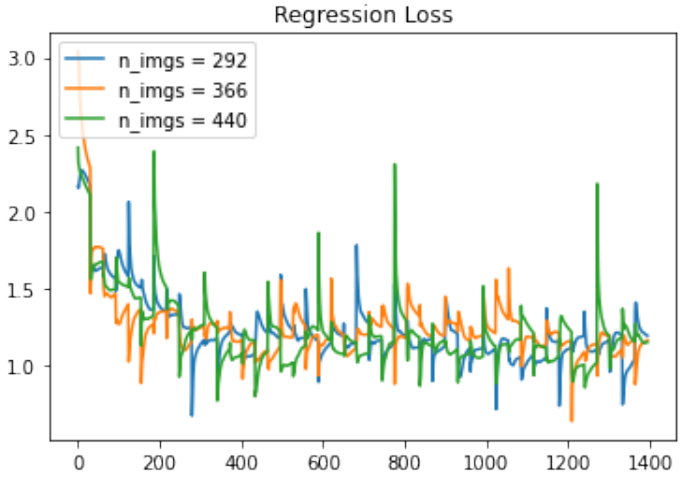
Fonte: Autor

Pelas figuras 18 e 19 pode-se considerar que alterações na quantidade de anotações no conjunto de imagens não alteram o desempenho da otimização. Neste caso, deve-se considerar que possivelmente, por se tratar um dataset relativamente pequeno as alterações não são suficientes para levar a modificações no desempenho dos algoritmos.

4.5.1.4 Aumentação de dados

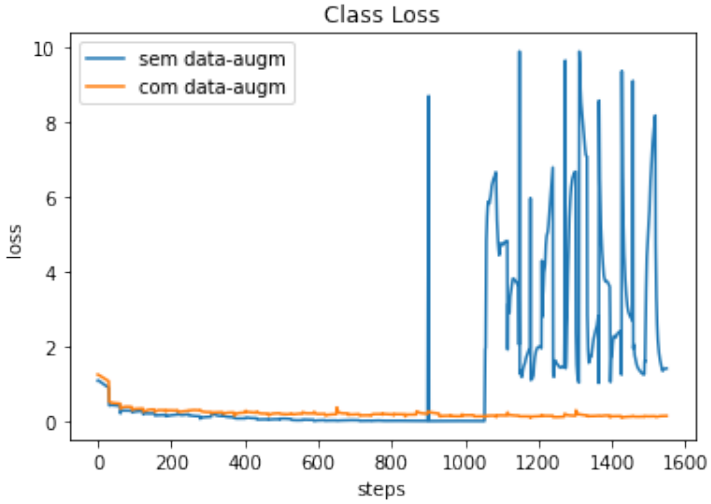
As figuras 20 e 21 a seguir apresentam as curvas de custo para os submodelos de classificação e regressão da arquitetura RetinaNet, com aumentação de dados e sem aumentação de dados. É usada a configuração padrão da implementação, e como parâmetros da chamada do método de treino: tamanho de batch = 8, taxa de aprendizado = 1e-5, quantidade de épocas = 30.

Figura 19 – Custo de Regressão para número de anotações diferente



Fonte: Autor

Figura 20 – Custo de Classificação para treinamento com e sem aumentação de dados



Fonte: Autor

Figura 21 – Custo de Regressão para treinamento com e sem aumentação de dados

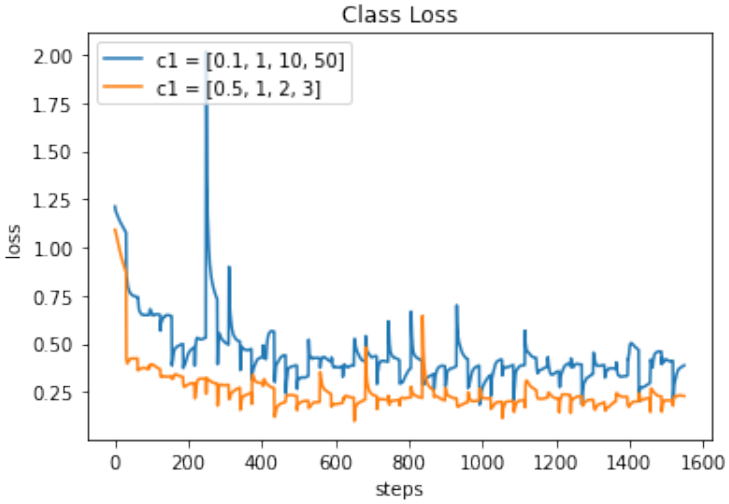
Fonte: Autor

Pelas figuras 20 e 21 verifica-se que ao utilizar aumentação de dados os algoritmos convergem e se mantém em seu valor mínimo. Por outro lado, quando não é utilizada, ocorre divergência em um determinado ponto, no caso da figura 20 e na figura 21 possivelmente ocorre overfitting já que o custo de treino continua decrescendo até o valor nulo.

4.5.1.5 Configurações de âncoras

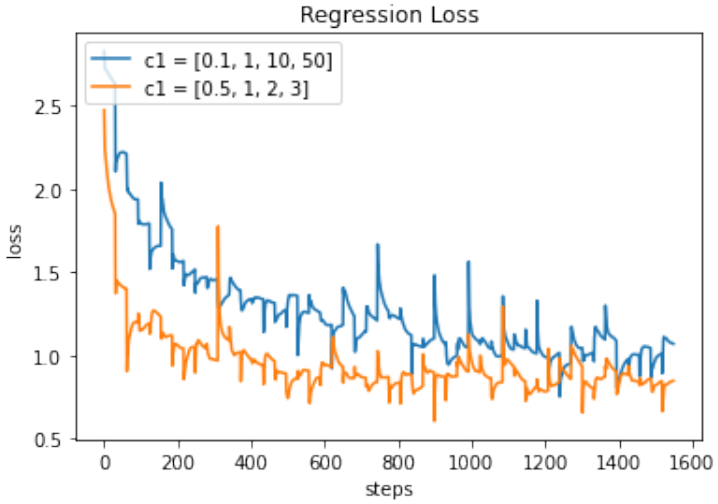
As figuras 22 e 23 a seguir apresentam as curvas de custo para os submodelos de classificação e regressão da arquitetura RetinaNet, utilizando diferentes razões de aspecto por localização de âncora. A modificação é feita através de configuração externa com arquivo de configuração. Excetuando-se as modificações nas configurações de âncoras, é utilizada a configuração padrão da implementação, e como parâmetros da chamada do método de treino: tamanho de batch = 8, taxa de aprendizado = 1e-4, quantidade de épocas = 50. As razões de aspecto são modificadas de c1 = [0.5, 1, 2, 3] para c1 = [0.1, 1, 10, 50].

Figura 22 – Custo de Classificação para modelo com diferentes razões de aspecto por âncora



Fonte: Autor

Figura 23 – Custo de Regressão para modelo com diferentes razões de aspecto por âncora



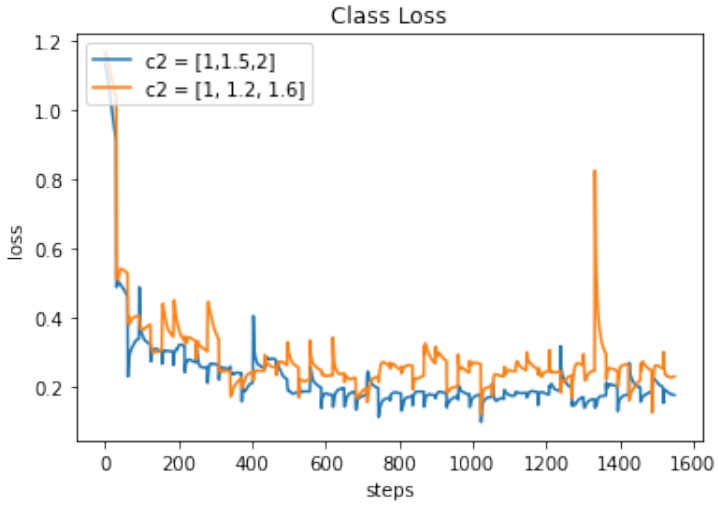
Fonte: Autor

Pelas figuras 22 e 23 pode-se considerar que alterações nas razões de aspecto por âncora podem levar a modificações no desempenho dos algoritmos. No caso apresentado, percebe-se que a configuração padrão para razões de aspecto por âncora leva a um desempenho melhor que a configuração proposta onde o intervalo de razões utilizadas é maior.

As figuras 24 e 25 a seguir apresentam as curvas de custo para os submodelos de classificação e regressão da arquitetura RetinaNet, utilizando diferentes fatores de escala por localização de âncora. A modificação é feita através de configuração externa com arquivo de configuração. Excetuando-se as modificações nas configurações de âncoras, é

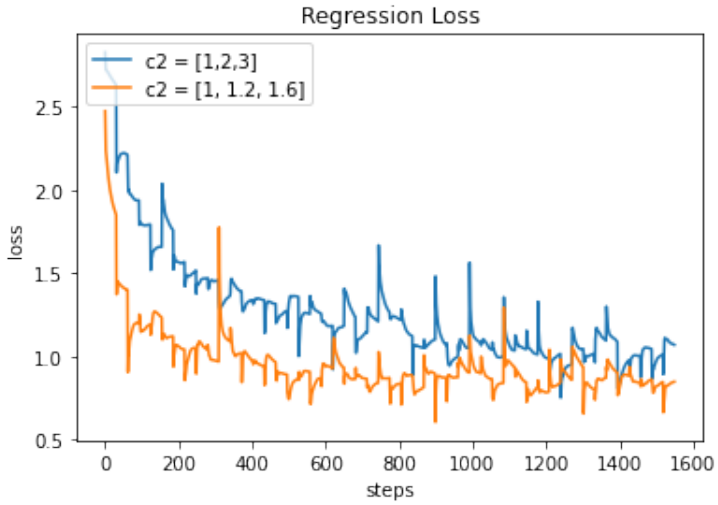
utilizada a configuração padrão da implementação, e como parâmetros da chamada do método de treino: tamanho de batch = 8, taxa de aprendizado = 1e-4, quantidade de épocas = 50. Os fatores de escala por localização de âncora são modificados de c2 = [1, 1.2, 1.6] para c2 = [1, 1.5, 2].

Figura 24 – Custo de Classificação para modelo com diferentes fatores de escala por localização de âncora



Fonte: Autor

Figura 25 – Custo de Regressão para modelo com diferentes fatores de escala por localização de âncora



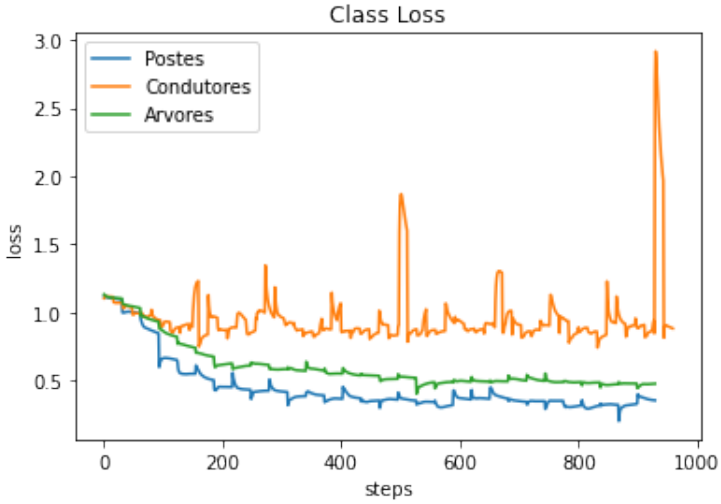
Fonte: Autor

Pelas figuras 24 e 25 pode-se considerar que a alteração aplicada na configuração dos fatores de escala não levou a grandes modificações no desempenho dos algoritmos.

4.5.1.6 Postes, Condutores e Árvores

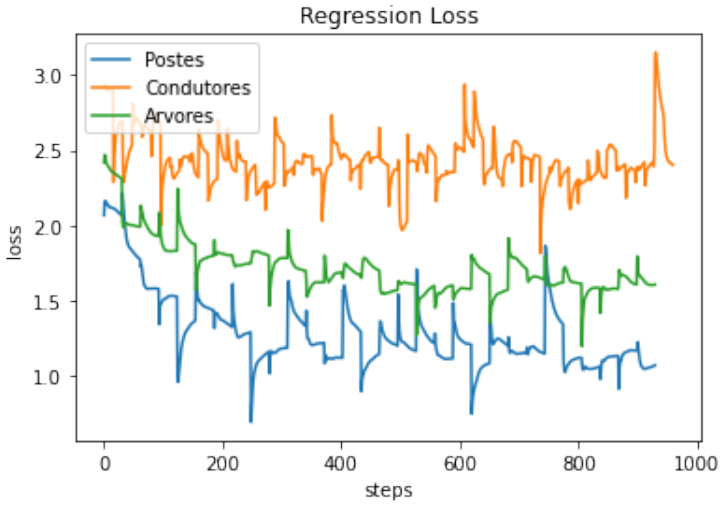
As figuras 26 e 27 a seguir apresentam as curvas de custo para os submodelos de classificação e regressão da arquitetura RetinaNet, para as diferentes classes de objetos. É utilizada a configuração padrão da implementação, com uso de aumentação de dados, e como parâmetros da chamada do método de treino: tamanho de batch = 8, taxa de aprendizado = 1e-4, quantidade de épocas = 30. Além disso, utiliza-se a mesma quantidade de imagens com anotações para todas = 125, limitada pelo dataset de condutores.

Figura 26 – Custo de Classificação para as diferentes classes



Fonte: Autor

Figura 27 – Custo de Regressão para as diferentes classes



Fonte: Autor

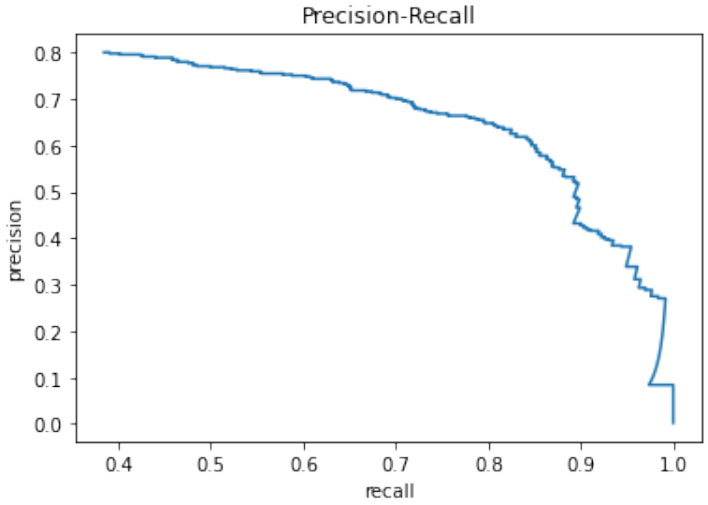
Pelas figuras 26 e 27 pode-se considerar que o desempenho dos algoritmos é influenciado pela classe de objetos utilizada, sendo o pior desempenho encontrado para a classe de condutores, e o melhor desempenho para a classe de postes. Vale observar que

no caso das árvores o desempenho observado pode ter sido pior por ter sido feita a divisão do total de anotações entre os diferentes tipos de árvores considerados.

4.5.1.7 Curva Precisão-Recall para Postes

A figura 28 a seguir apresenta a curva de precisão x recall para o submodelo de classificação da arquitetura RetinaNet, para imagens da classe de postes e utilizando NMS = 0.17. É utilizada a configuração padrão da implementação, com uso de aumentação de dados, e como parâmetros da chamada do método de treino: tamanho de batch = 10, taxa de aprendizado = 1e-3, quantidade de épocas = 30.

Figura 28 – Curva Precisão - Recall da Classificação



Fonte: Autor

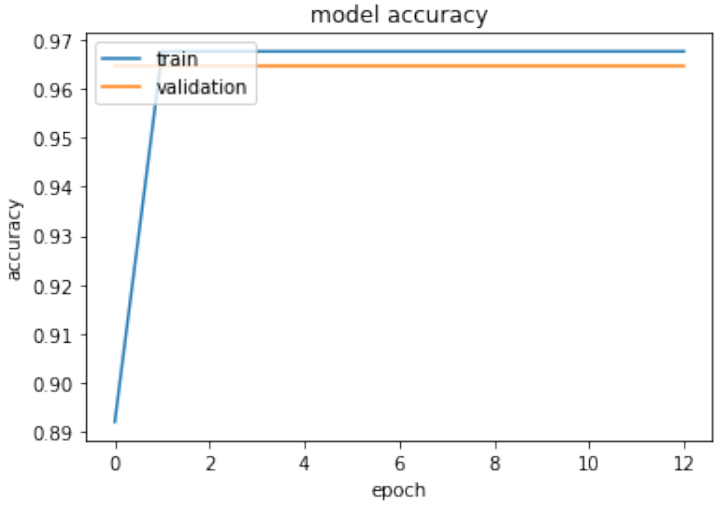
Para a curva Precisão-Recall apresentada, tem-se associado um valor mAP = 0.7253.

4.5.2 Segmentação Semântica (U-Net)

4.5.2.1 Comparação entre modelos

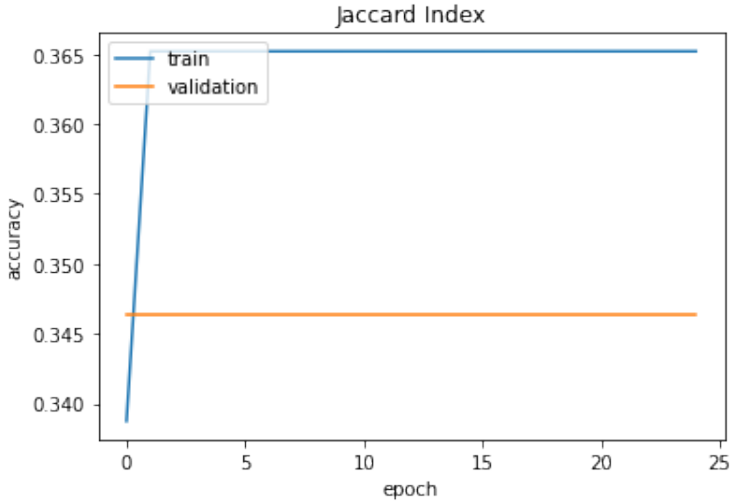
As figuras 29 e 30 a seguir apresentam as curvas de métricas de interesse distintas, considerando definições de função de custo distintas (Entropia Cruzada Binária, BCE, e Distância de Jaccard), aplicadas ao problema de segmentação com a arquitetura U-Net. Foram utilizados como parâmetros no treinamento do modelo: tamanho de batch = 1, número de épocas = 25, taxa de aprendizado = 1e-2, divisão do conjunto de imagens em treino e teste com proporção 80%-20% respetivamente, e algoritmo de otimização Adam.

Figura 29 – Acurácia para Segmentação Semântica em treino e validação



Fonte: Autor

Figura 30 – Índice de Jaccard (IoU) para Segmentação Semântica em treino e validação



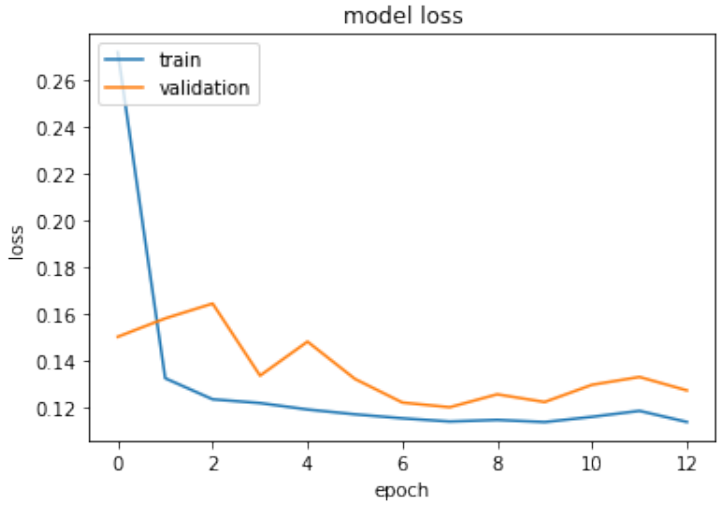
Fonte: Autor

Pelas figuras 29 e 30 percebe-se que a métrica acurácia binária, que está relacionada com a aplicação da função de custo entropia cruzada binária, apresentou valores significativamente maiores do que o do índice de Jaccard, associado com a função de custo distância de Jaccard, sendo que ambas estão situadas no intervalo entre 0 e 1. No entanto, deve-se levar em conta que a métrica índice de Jaccard fornece uma descrição mais precisa para o desempenho do algoritmo visto que não é prejudicada pelo desbalanceamento entre as classes.

As figuras 31 e 32 a seguir apresentam as curvas de custo, para definições de função de custo distintas (BCE e Distância de Jaccard), aplicadas ao problema de segmentação com a arquitetura U-Net. Foram utilizados como parâmetros no treinamento do modelo:

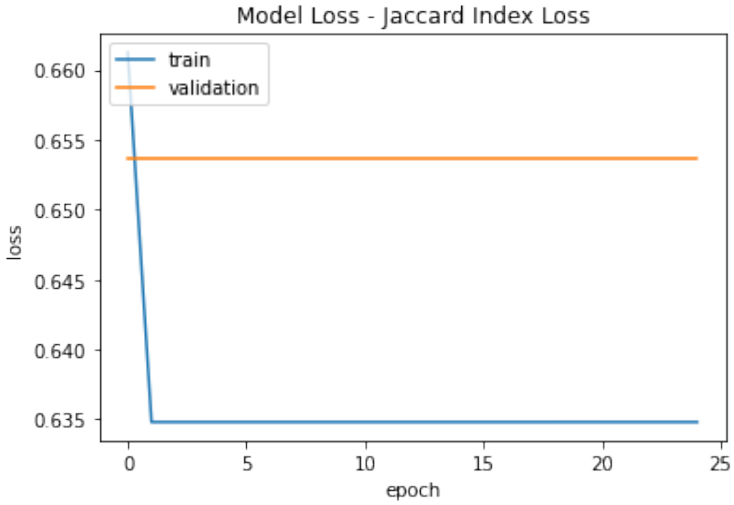
tamanho de batch = 1, número de épocas = 12 para BCE e número de épocas = 25 para Distância de Jaccard, taxa de aprendizado = 1e-2, divisão do conjunto de imagens em treino e teste com proporção 80%-20% respetivamente, e algoritmo de optimização Adam.

Figura 31 – Curva de custo para Segmentação Semântica em treino e validação utilizando função BCE



Fonte: Autor

Figura 32 – Custo Distância de Jaccard para Segmentação Semântica em treino e validação



Fonte: Autor

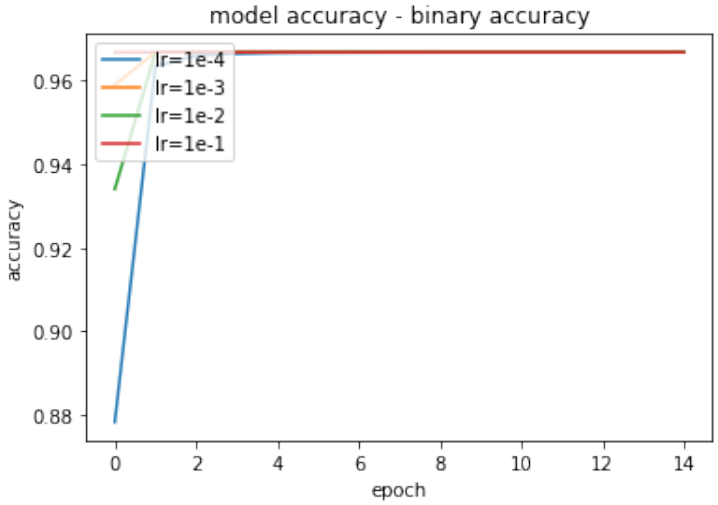
Pelas figuras 31 e 32 percebe-se que a ção cuseo entropia cruzada binária, apresentou valores significativamente menores que a distância de Jaccard. No entanto, da mesma forma que para as métricas, deve-se levar em conta que a distância de Jaccard fornece uma descrição mais precisa para o desempenho do algoritmo por não ser prejudicada pelo desbalanceamento entre as classes.

Pelas observações de ambas métricas e funções de custo, percebe-se que o desempenho alcançado na tarefa de segmentação foi relativamente baixo.

4.5.2.2 Taxa de Aprendizado

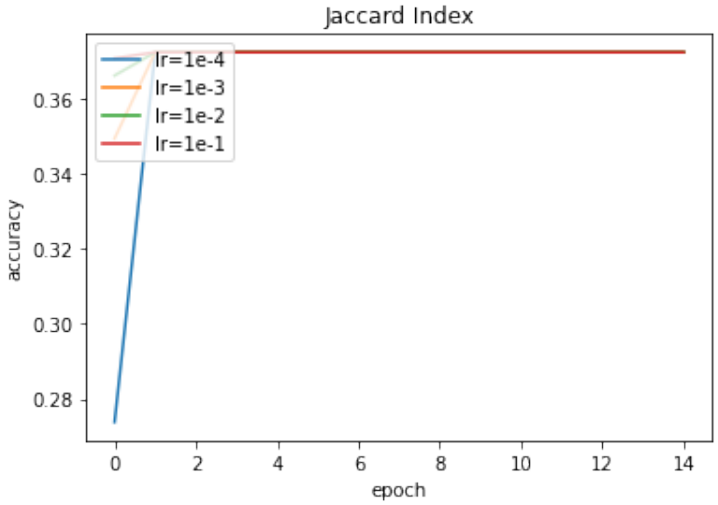
As figuras 33 e 34 a seguir apresentam as curvas das métricas de interesse (acurácia e índice de Jaccard) para segmentação semântica com a arquitetura U-Net utilizando diferentes taxas de aprendizado. São utilizados como parâmetros: tamanho de batch = 8, quantidade de épocas = 30.

Figura 33 – Acurácia para Segmentação Semântica utilizando função de custo BCE para diferentes taxas de aprendizagem



Fonte: Autor

Figura 34 – Acurácia para Segmentação Semântica utilizando função Distância de Jaccard para diferentes taxas de aprendizagem

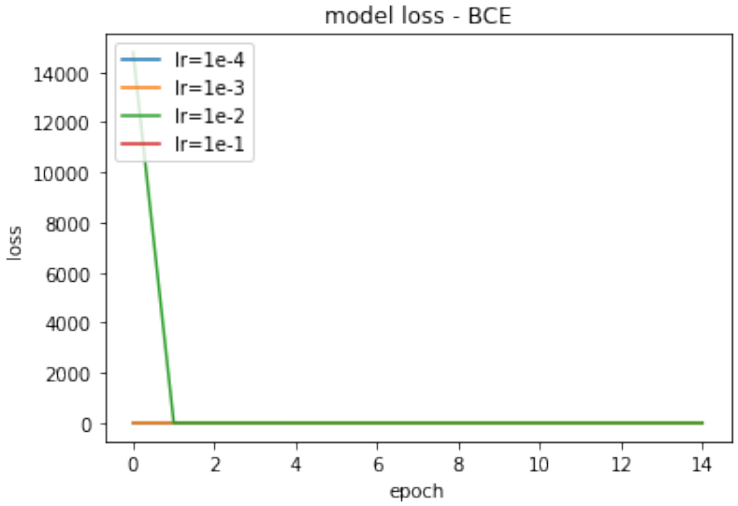


Fonte: Autor

Portanto, pelas figuras 33 e 34 pode-se considerar que alterações na taxa de aprendizado tiveram pouca influência nos valores alcançados para métricas.

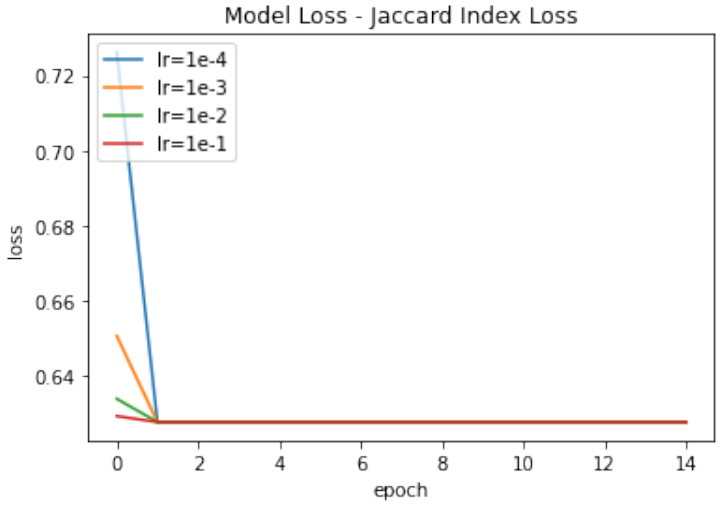
As figuras 35 e 36 a seguir apresentam as curvas de custo para segmentação semântica com a arquitetura U-Net utilizando diferentes taxas de aprendizado. São utilizados como parâmetros: tamanho de batch = 8, quantidade de épocas = 30.

Figura 35 – Custo para Segmentação Semântica utilizando diferentes taxas de aprendizagem para função BCE



Fonte: Autor

Figura 36 – Acurácia para Segmentação Semântica utilizando função Distância de Jaccard para diferentes taxas de aprendizagem



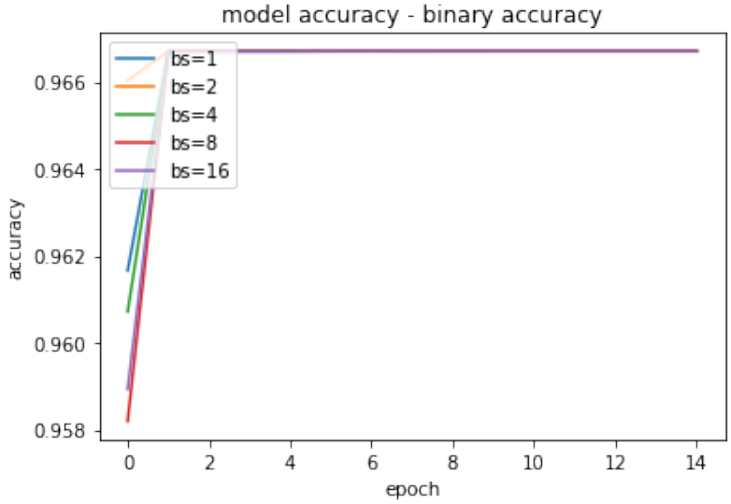
Fonte: Autor

Da mesma forma que para as métricas, pelas figuras 33 e 34 pode-se considerar que alterações na taxa de aprendizado tiveram pouca influência nos valores alcançados e também na velocidade de convergência de ambas as funções de custo.

4.5.2.3 Batch Size

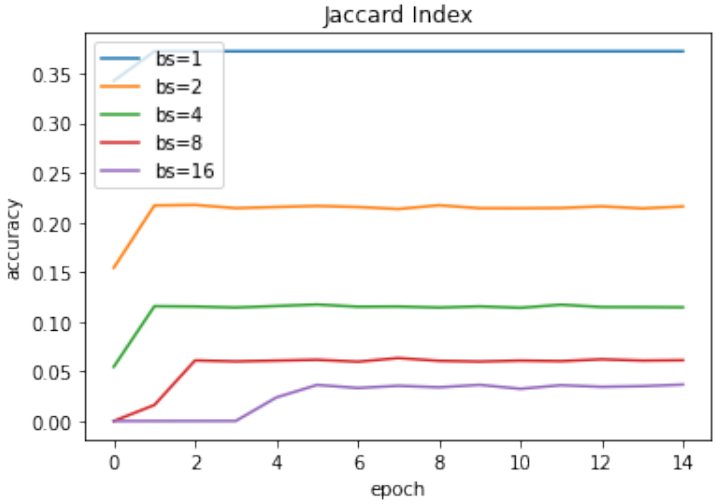
As figuras 37 e 38 a seguir apresentam as curvas das métricas de interesse (acurácia e índice de Jaccard) para segmentação semântica com a arquitetura U-Net utilizando tamanhos de batch diferentes. São utilizados como parâmetros: taxa de aprendizado = 1e-3 (padrão para o otmizador Adam do framewokr Keras), quantidade de épocas = 30.

Figura 37 – Acurácia para Segmentação Semântica utilizando função de custo BCE para tamanhos de batch diferentes



Fonte: Autor

Figura 38 – Acurácia para Segmentação Semântica utilizando função Distância de Jaccard para tamanhos de batch diferentes



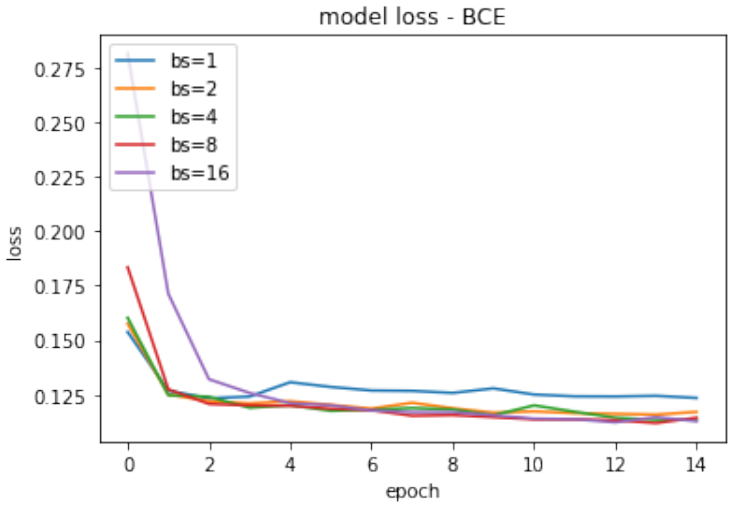
Fonte: Autor

Portanto, pelas figuras 37 e 38 pode-se considerar que alterações no tamanho de batch influenciam os valores alcançados para a métrica índice de Jaccard. Observa-se que

neste caso que o uso de tamanho de batchs menores leva a melhores resultados, alcançado o melhor valor para tamanho de batch = 1.

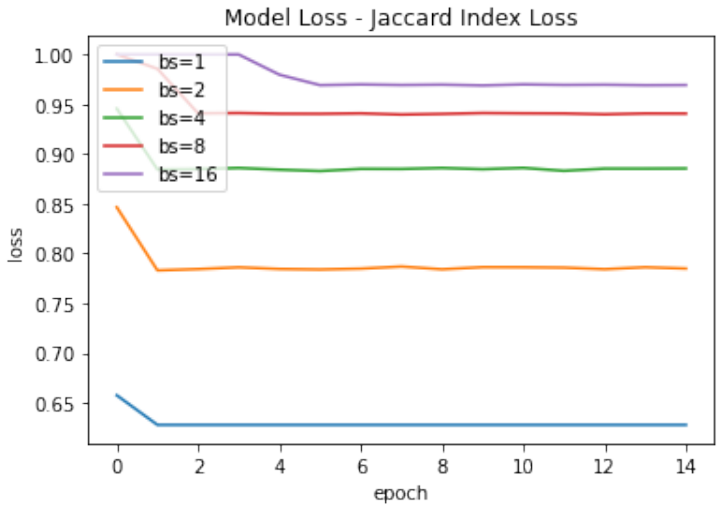
As figuras 39 e 40 a seguir apresentam as curvas do custo para segmentação semântica com a arquitetura U-Net utilizando tamanhos de batch diferentes. São utilizados como parâmetros: taxa de aprendizado = 1e-3 (padrão para o otmizador Adam do framewokr Keras), quantidade de épocas = 30.

Figura 39 – Custo para Segmentação Semântica utilizando tamanhos de batch diferentes utilizando a função BCE



Fonte: Autor

Figura 40 – Acurácia para Segmentação Semântica utilizando função Distância de Jaccard para tamanhos de batch diferentes



Fonte: Autor

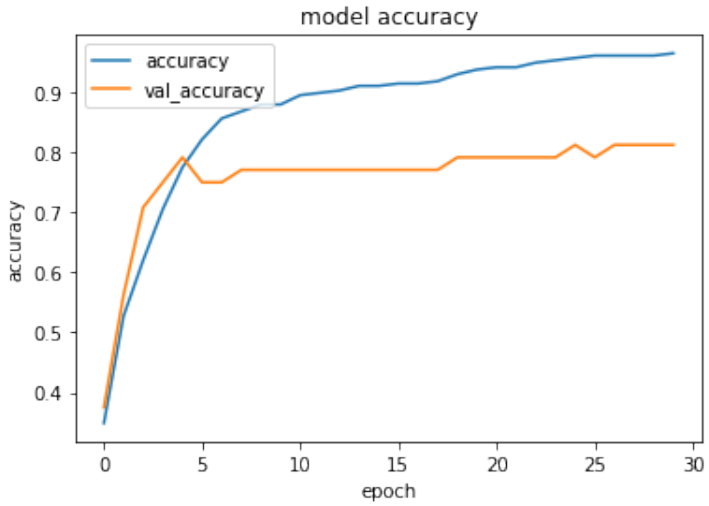
Da mesma forma que para as métricas, pelas figuras 39 e 40 verifica-se que alterações no tamanho de batch influenciam nos valores alcançados pelas funções custo. Sendo que,

neste caso, tanto a otimização da função entropia cruzada binária como a distância de Jaccard são modificados. No caso da entropia cruzada binária o maior tamananho de batch, bs = 16, levou a uma convergência mais lenta, enquanto o menor tamanho de batch, bs = 1 levou a um valor final maior, indicando desempenho pior.

4.5.2.4 Classificação de espécies de árvores

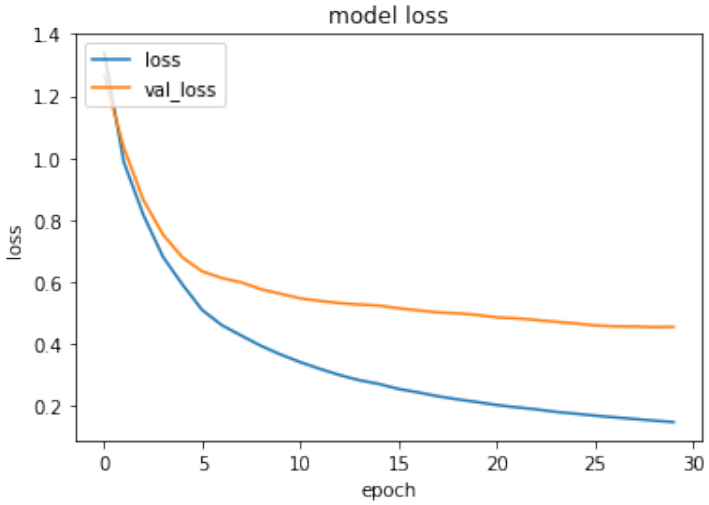
As figuras 41 e 42 a seguir apresentam as curvas do custo e acurácia para classificação da tarefa de classificação de espécies. São utilizados como parâmetros da chamada taxa de aprendizado = 1e-3, tamanho de bacth do ImageDataGenerator = 100 e quantidade de épocas = 30.

Figura 41 – Acurácia para classficação das 3 espécies de árvores



Fonte: Autor

Figura 42 – Custo para classficação das 3 espécies de árvores



Fonte: Autor

Pelas figuras 41 e 42 é verificado um bom desempenho para a arquitetura utilizada na tarefa de classificação, considerando as 3 espécies utilizadas, para o dataset produzido através de imagens da GBIF.

4.5.3 Síntese da Metodologia Utilizada

A seguir é apresentada a síntese da metodologia utilizada:

1. a) Módulo para obtenção das imagens GSV
   b) Módulo para obtenção das imagens GBIF
2. a) Ferramentas para rotulação de imagens do GSV:
      - LabelImg para Caixas delimitadoras
      - Labelme para áreas segmentadas
   b) Algoritmo para clusterização das imagens GBIF
3. Algoritmos de processamento de imagens com DL:
   - Resnet para classificação de espécies de árvores
   - RetinaNet, para detecção de condutores, postes e árvores
   - U-net, para segmentação de condutores
4. Módulo para integração de algoritmos e aplicação das imagens de teste
   - Integração dos modelos de detecção de objetos (condutores, postes e árvores)
   - Integração de algoritmo para detecção de árvores e classificação de espécies

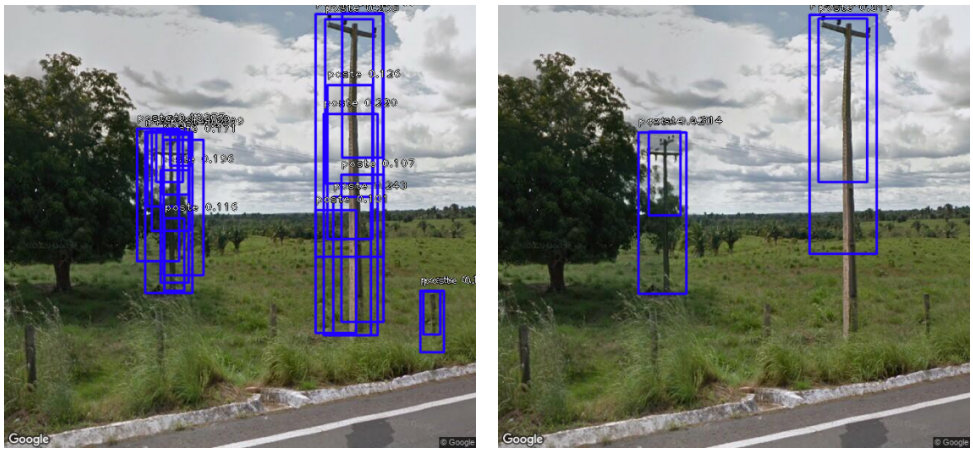
5 Resultados e Discussão

Tendo realizado um conjunto de testes com os diferentes parâmetros do modelo, verificando a influência de cada um deles, e determinando aqueles que melhores se aplicam a cada uma das tarefas. A seguir serão apresentados alguns dos resultados obtidos para as tarefas consideradas.

5.1 Detecção de Postes

As figuras 43.a e 43.b a seguir apresentam resultados da aplicação do modelo treinado para detecção de postes na inferência das caixas delimitadoras em uma imagem do conjunto de testes, contendo postes.

Figura 43 – Resultados para detecção de postes com Retinanet



(a) Baixo valor NMS (b) Alto valor NMS

Conforme observado, a detecção pode ser realizada. No entanto, para resultados mais precisos deve ser feito um ajuste do valor de supressão de não-máximos (NMS), de modo que apenas as maiores pontuações sejam consideradas predições corretas. Assim, na figura 43.b é utilizado um valor de NMS maior restringindo o conjunto de caixas delimitadoras aceitas.

5.2 Detecção de Condutores

As figuras 44.a e 44.f a seguir apresentam resultados da aplicação do modelo treinado para detecção de condutores na inferência das caixas delimitadoras em diferentes imagens do conjunto de testes, contendo condutores.

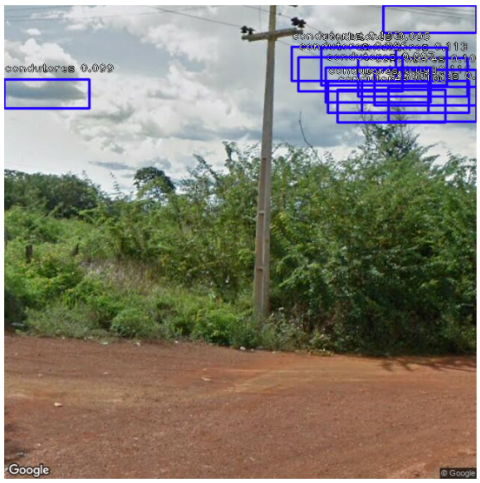
Figura 44 – Resultados para detecção de condutores com Retinanet

(a) Vista 1

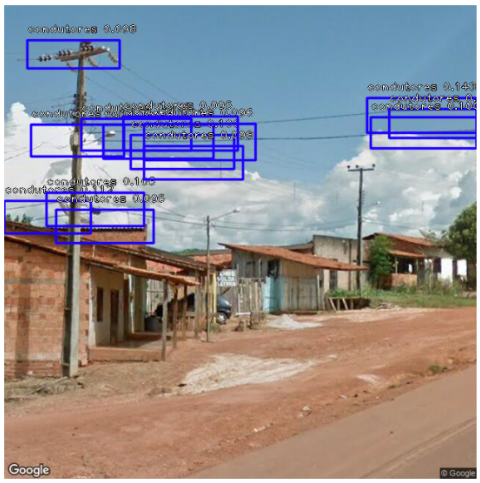
(b) Vista 2

(c) Vista 3

(d) Vista 4



(e) Vista 5



(f) Vista 6

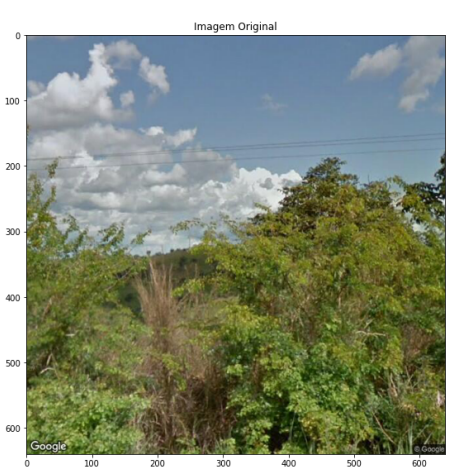
Pelas figuras, observam-se as restrições do uso do algoritmo para esta classe de objetos. Isso porque, para que seja feita a cobertura dos condutores existe grande redundância, com muitas caixas delimitadoras cobrindo um mesmo trecho a partir de pontos distintos. Fora isso, o controle sobre a quantidade de caixas delimitadoras se torna difícil e impreciso.

5.3 Segmentação dos condutores

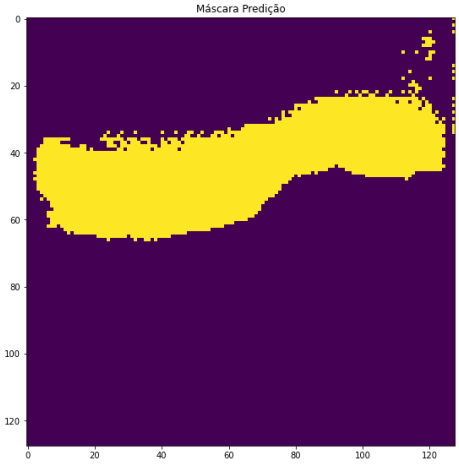
Pela descrição da metodologia e confirmação pelos resultados anteriores, o uso da arquitetura Retinanet possui limitações na detecção de condutores.

A alternativa proposta é utilizar segmentação de imagens. Portanto, as figuras 45.a e 45.b a seguir apresentam resultados do modelo de segmentação semântica treinado, aplicado no processo de inferência das segmentações em uma imagem do conjunto de testes, contendo condutores. A figura 45.b está associada com acurácia binária = 0.9648, obtida pela otimização da entropia cruzada binária como função custo para o modelo.

Figura 45 – Resultados para segmentação de condutores, obtidos com treinamento da função custo Entropia Cruzada Binária



(a) Imagem Original



(b) Máscara gerada através de predição

Conforme observado, a segmentação pôde ser realizada, com limitações, para a função custo BCE. Por outro lado, isto não foi possível através da função custo Distância de Jaccard. Deve-se observar entretanto que estes resultados podem não ter uma interpretação diretamente relacionada com os valores obtidos nas curvas testes. Isso porque o uso da acurácia binária e entropia cruzada binária como métrica e função custo para representação deste problema tem deficiências como observado em 3.7.2.

A respeito do desempenho ruim da métrica índice de Jaccard, conforme descrito nos testes da metodologia, possivelmente é devido às limitações encontradas nas configurações

do treinamento, visto que é utilizado um conjunto de imagens relativamente pequeno, e diferente da Retinanet, neste caso não é utilizada transferência de aprendizado.

Deve ser ressaltado ainda que o desempenho observado na figura 45 encontra-se entre alguns dos melhores para a configuração citada. Tendo sido mostrado para a confirmação dos objetivos da metodologia. No entanto, para outras situações, o desempenho qualitativo ainda é ruim. Assim, acredita-se que o método seja adequado para a metodologia proposta, no entanto, deverá ser refinado futuramente para obtenção de resultados mais precisos.

5.4 Agrupamento das imagens de espécies de árvores

As figuras 46.a e 46.c a seguir apresentam alguns dos agrupamentos obtidos pela aplicação do K-means aos conjuntos de imagens de espécies de árvores:

Figura 46 – Resultados da aplicação do algoritmo K-means ao conjunto de imagens do GBIF

(a) Agrupamento 1 para espécie de árvore Anacardium

(b) Agrupamento 2 para espécie de árvore Anacardium

(c) Agrupamento 3 para espécie de árvore Anacardium

É possível observar que em cada agrupamento temos conjuntos de características particulares. No agrupamento da figura 46.a temos imagens com similaridade com as imagens do GSV, ou seja, imagens de vista frontal e que incluem uma visão global da copa da árvore. Por outro lado, no agrupamento da figura 46.b as imagens possuem foco nos frutos, e no agrupamento da figura 46.c as imagens possuem foco nas folhas, estas são algumas das características possíveis, no entanto, pode haver ainda agrupamentos com imagens de baixa qualidade, contendo elementos que não representam características de espécies de árvores.

De qualquer modo, é verificado que o uso do GBIF é adequado para a tarefa de obtenção de um dataset com imagens que podem ser utilizadas no sistema proposto, em

especial no treinamento de um algoritmo de classificação de espécies para as imagens do GSV.

5.5 Detecção de Árvores

As figuras 47.a a 47.f a seguir apresentam resultados da aplicação do modelo treinado para detecção de árvores na inferência das caixas delimitadoras em uma imagem do conjunto de testes, contendo árvores.

Figura 47 – Resultados para detecção de árvores

(a) Detecção indicando árvores tipo 1 e 2

(b) Detecção indicando árvore tipo 2

(c) Detecção indicando árvores tipo 4

(d) Detecção indicando árvores tipo 3

É observado que o algoritmo pode apresentar desempenho diferente de acordo com o tipo de árvore presente. Assim para a árvore de "Tipo 4"a pontuação do algoritmo é maior levando a uma quantidade de propostas maior. Tal observação faz sentido considerando o

desbalanceamento entre as ocorrências dos tipos de árvores. Outro fator que pode produzir este resultado é a presença de características específicas de cada tipo. Em especial, se considerarmos o "Tipo 4"que apresenta características facilmente observáveis, ou seja, que se diferenciam com maior facilidade em relação aos outros, mesmo que seja dada capacidade de generalização ao algoritmo, possivelmente poderá manter seu desempenho, enquanto que para os outros tipos onde as diferenças não são tão nítidas, o desempenho estará relacionado a maior especificidade, caso isso não ocorra haverá classificações incorretas entre tipos, ou até mesmo uma dupla classificação como é mostrado na figura (a), onde as pontuações das classes "tipo 1"e "tipo 2"para a mesma figura é próxima.

5.6 Integração dos Métodos

Com a definição de métodos que permitem fazer a identificação de elementos da vegetação e elementos da infraestrutura do sistema de distribuição de energia, é necessário ainda que seja analisada a interação entre estes elementos e, além disso, que se tenha uma ferramenta que possa ser utilizada pelos operadores e gestores dos sistemas de distribuição de energia. Deste modo, é realizada a integração dos resultados dos diferentes modelos de detecção de objetos (detecção para condutores, postes e árvores) e integração entre detecção de árvores e classificação de espécies de árvores. O sistema integrado deve ser aplicado ao conjunto de todas as coordenadas dos elementos da infraestrutura disponíveis. A integração entre os modelos permitirá que seja realizada uma análise da periculosidade de cada ativo a partir das características visuais, em especial o posicionamento dos objetos de interesse na imagem. Com isso será possível avaliar problemas como invasão de vegetação, através da proximidade das árvores com os ativos, ou, para critérios mais específicos, se as distâncias entre vegetação e estes ativos estão nos intervalos permitidos pelas normas do setor elétrico, definindo a necessidade ou não da realização de podas. Finalmente, a aplicação do sistema integrado ao conjunto de coordenadas possibilitará a criação de mapas contendo estatísticas e indicadores sobre a periculosidade das regiões, auxiliando a tomada de decisão em cenários mais gerais, e de acordo com características espaciais.

Considerando os aspectos citados e, portanto, a integração dos métodos, é gerado um modelo de dados com capacidade de representação dos diferentes elementos de interesse, assim como do posicionamento destes elementos nas imagens. O modelo de dados é representado na tabela 4 a seguir:

Tabela 4 – Saída dos métodos integrados

	xmin	ymin	xmax	ymax	Score	Classe	Vista	Espécie
Coorden.: 531	0.000	1.915	6.203	19.920	0.054	Poste	Frontal	
Coorden.: 531	0.000	0.000	20.320	20.320	0.101	Arvore	Frontal	cocos-nucifera
Coorden.: 548	0.000	0.000	20.320	18.261	0.118	Arvore	Frontal	mangifera-indica
Coorden.: 549	0.000	0.000	9.918	19.914	0.064	Poste	Frontal	
Coorden.: 549	2.966	8.704	20.320	20.320	0.131	Arvore	Frontal	cocos-nucifera
Coorden.: 493	0.000	0.000	19.392	8.499	0.098	Condutor	Frontal	
Coorden.: 448	1.139	0.137	20.320	19.286	0.140	Arvore	Frontal	mangifera-indica

Portanto, são produzidas informações adequadas para a análise e planejamento de podas, assim como para a verificação da periculosidade de cenários mais específicos. No entanto, o interesse é que esta análise fosse aplicada com diferentes ângulos de orientação horizontal, ou ao menos em vista frontal e ortogonal, permitindo obter uma caracterização espacial mais precisa. Além disso, não são apresentadas métricas de avaliação dos detectores. Em vista das limitações de tempo para o desenvolvimento do trabalho, estas informações não puderam ser adquiridas.

6 Conclusões e Trabalhos Futuros

6.1 Conclusões

Observa-se que o sistema cumpre parte dos objetivos de interesse deste trabalho, fornecendo uma metodologia automática baseada em técnicas de aprendizagem de máquina e processamento de imagem que poderá ser utilizada na análise e planejamento de poda, assim como na verificação da periculosidade de cenários mais específicos. No entanto, em vista das limitações de tempo para o desenvolvimento do trabalho, algumas das análises de interesse não puderam ser desenvolvidas.

Em segundo lugar, conclui-se que o uso do GSV como ferramenta para aquisição de imagens pode ser interessante para estudos que visem a localização de postes e de de árvores. No entanto, devem ser consideradas limitações no seu uso, em especial no monitoramento de condutores. Isso porque apenas em alguns tipos de vistas e cenários é possível fazer a observação destes elementos com boa resolução. Da mesma forma, o uso do GBIF como ferramenta para aquisição de imagens das espécies de árvores mostrou desempenho satisfatório, com acurácia de validação de 0.8, na definição de um método de classificação para as espécies selecionadas.

A respeito das arquiteturas de redes convolucionais, foi observado que a RetinaNet apresenta desempenho satisfatório com mAP de 0.7253 para detecção de objetos da classe postes. No entanto, o desempenho depende das classes de analisadas, sendo pior para árvores e condutores. Por outro lado, a arquitetura U-Net não apresentou um desempenho tão bom mesmo com a definição de diferentes funções custo. Este pode ter sido um resultado do desbalanceamento entre as classes, no caso da função custo entropia cruzada binária, visto que sua acurácia de validação chegou ao valor 0.96, e também da baixa quantidade de imagens, e qualidade relativamente baixa das mesmas, dificultando a tarefa de anotação para a classe de condutores. No caso da função custo Distância de Jaccard, que obteve acurácia de validação de 0.34, o desempenho pode estar relacionado principalmente da baixa quantidade e qualidade das imagens. Deve-se observar que a quantidade de anotações também foi uma limitação na detecção de objetos.

Apesar das restrições citadas, e embora não tenham sido realizados testes para todos as classes de objetos, pode-se considerar a detecção de objetos um método eficaz a ser aplicado. Isso porque, ao realizar as detecções, são fornecidas diretamente as classes e posicionamento dos elementos de intesse, sendo estas as principais informações para a realização das análises. Por outro lado, acredita-se que a tarefa de segmentação também possa ser utilizada, podendo gerar todo o conjunto de posições que um objeto ocupa,

garantindo análises mais precisas sobre a proximidade de elementos dentro da imagem.

6.2 Trabalhos Futuros

Para trabalhos futuros, é interessante que seja realizada uma quantidade maior de testes, e que sejam definidos critérios mais específicos para a avalição das tarefas. Neste sentido, algumas análise de interesse são:

- Analisar datas de imagens e taxa de crescimento das espécies.
- Utilizar indicadores de confiabilidade dos sistemas de distribuição de energia como DEC e FEC [47].

Também é possível que sejam utilizadas arquiteturas e metodologias mais recentes da área de processamento de imagens com DL. Entre elas a segmentação de instâncias possibilitará a realização de segmentação, garantindo informações de posicionamento mais precisas além de também apresentar as classes de interesse.

Para a tarefa de detecção de condutores vale considerar ainda a possibilidade de estudo e aplicação de alternativas específicas, como o modelo *R3Det* apresentado em [48], que adapta a arquitetura Retinanet possibilitando a detecção de objetos rotacionados, levando em conta a rotação das caixas delimitadoras. Outra alternativa é a arquitetura *LS-Net* apresentada em [49] com o objetivo de realizar a detecção os condutores (linhas de transmissão) com um detector de segmentos de linha de um único de estágio. A arquitetura utiliza um extrator de características através de rede convolucional, um classificador e um algoritmo para regressão de segmentos de linhas, sendo treinada com anotações geradas de forma sintética para fazer o treinamento.

Referências

[1] W. Zhang, C. Witharana, W. Li, C. Zhang, X. Li, and J. Parent, "Using deep learning to identify utility poles with crossarms and estimate their locations from google street view images," *Sensors*, vol. 18, no. 8, p. 2484, 2018. Citado 5 vezes nas páginas 4, 7, 9, 25 e 38.

[2] H. G. F. d. Lima, "Identificação e estimativa da altura de árvores em imagens de satélite e do google street view," 2016. Citado 6 vezes nas páginas 5, 1, 3, 4, 18 e 38.

[3] A. K. Lopes, "Reconhecimento de postes da rede elétrica em vias urbanas em imagens do google street view," 2016. Citado 3 vezes nas páginas 5, 4 e 38.

[4] COPEL, *Guia de Arborização*. Acessado em Março de 2021. Citado 2 vezes nas páginas 5 e 37.

[5] A. Moura, A. Moura, and E. Rocha, *Transmissão de Energia Elétrica em Corrente Alternada*. 2019. Citado 2 vezes nas páginas 1 e 33.

[6] G. D. N. Velasco, *Arborização viária X sistemas de distribuição de energia elétrica: avaliação dos custos, estudo das podas e levantamento de problemas fitotécnicos*. PhD thesis, Universidade de São Paulo, 2003. Citado na página 1.

[7] R. Jenssen, D. Roverso, and V. N. Nguyen, "Automatic autonomous vision-based power line inspection: A review of current status and the potential role of deep learning," *International Journal of Electrical Power & Energy Systems*, vol. 99, pp. 107–120, 2018. Citado 6 vezes nas páginas 1, 5, 6, 18, 25 e 38.

[8] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, vol. 1. MIT press Cambridge, 2016. Citado 9 vezes nas páginas 1, 14, 15, 17, 18, 19, 20, 21 e 22.

[9] S. Branson, J. D. Wegner, D. Hall, N. Lang, K. Schindler, and P. Perona, "From google maps to a fine-grained catalog of street trees," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 135, pp. 13–30, 2018. Citado 2 vezes nas páginas 8 e 38.

[10] P. Norvig and S. Russell, *Inteligencia artificial*, vol. 3ª edição. 2013. Citado 3 vezes nas páginas 11, 15 e 16.

[11] H. Daumé III, "A course in machine learning," *Publisher, ciml. info*, vol. 5, p. 69, 2012. Citado 4 vezes nas páginas 12, 15, 16 e 17.

[12] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, vol. 112. Springer, 2013. Citado na página 12.

[13] S. Haykin, *Redes neurais: princípios e prática*. Bookman Editora, 2007. Citado na página 17.

[14] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016. Citado na página 17.

[15] A. Lee, "Comparing deep neural networks and traditional vision algorithms in mobile robotics," *Swarthmore University*, 2015. Citado na página 18.

[16] G. Lindsay, "Convolutional neural networks as a model of the visual system: past, present, and future," *Journal of Cognitive Neuroscience*, pp. 1–15, 2020. Citado na página 19.

[17] M. A. Nielsen, *Neural networks and deep learning*, vol. 25. Determination press San Francisco, CA, 2015. Citado 3 vezes nas páginas 19, 20 e 21.

[18] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola, "Dive into deep learning. 2020," 2020. Citado 5 vezes nas páginas 20, 21, 22, 23 e 30.

[19] J. Brownlee, "A gentle introduction to batch normalization for deep neural networks." Acessado em Março de 2021. Citado na página 22.

[20] Keras, "Transfer learning fine-tuning." Acessado em Março de 2021. Citado 2 vezes nas páginas 22 e 23.

[21] M. Z. Alom, T. M. Taha, C. Yakopcic, S. Westberg, P. Sidike, M. S. Nasrin, M. Hasan, B. C. Van Essen, A. A. Awwal, and V. K. Asari, "A state-of-the-art survey on deep learning theory and architectures," *Electronics*, vol. 8, no. 3, p. 292, 2019. Citado 3 vezes nas páginas 23, 24 e 25.

[22] J. Brownlee, "How to develop vgg, inception and resnet modules from scratch in keras." Citado na página 24.

[23] J. Bertels, T. Eelbode, M. Berman, D. Vandermeulen, F. Maes, R. Bisschops, and M. B. Blaschko, "Optimizing the dice score and jaccard index for medical image segmentation: Theory and practice," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pp. 92–100, Springer, 2019. Citado 2 vezes nas páginas 26 e 31.

[24] B. Kayalibay, G. Jensen, and P. van der Smagt, "Cnn-based segmentation of medical imaging data," *arXiv preprint arXiv:1701.03056*, 2017. Citado na página 26.

[25] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017. Citado 6 vezes nas páginas 26, 27, 28, 29, 41 e 42.

[26] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015. Citado 3 vezes nas páginas 26, 27 e 43.

[27] A. Rosebrock, "Opencv selective search for object detection." Citado na página 27.

[28] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017. Citado na página 28.

[29] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440, 2015. Citado na página 30.

[30] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015. Citado 3 vezes nas páginas 30, 31 e 44.

[31] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017. Citado na página 30.

[32] H. Y. Kim, "Materiais do curso - pee-5796 algoritmos para processamento, análise e síntese de imagens e aprendizagem de máquinas," 2020. Citado na página 30.

[33] M. A. Rahman and Y. Wang, "Optimizing intersection-over-union in deep neural networks for image segmentation," in *International symposium on visual computing*, pp. 234–244, Springer, 2016. Citado na página 31.

[34] ABRADEE, "Redes de energia elÉtrica." Acessado em Março de 2021. Citado na página 33.

[35] C. L. Creder, Hélio, *Instalações Elétricas*, vol. 16ª edição. 2016. Citado na página 33.

[36] "Materiais do curso sdee - aspectos gerais do sistema de distribuição." Citado 2 vezes nas páginas 33 e 34.

[37] CEMIG, *Manual de Distribuição Projetos de Redes de Distribuição Aéreas Urbanas*. Citado na página 33.

[38] CELG, *Norma Técnica de Distribuição - "Ferragens para Redes Aéreas de Distribuição de Energia Elétrica- Especificação e Padronização (CELG)*. Citado na página 33.

[39] Elektro, *Elektro - "Redes Protegidas Compactas - Critérios para Projetos e Padronização de Estruturas*

. Citado na página 34.

[40] Rede Energia - Energisa, *NORMA TÉCNICA DE DISTRIBUIÇÃO NTD-RE-001 MONTAGEM DE REDES DE DISTRIBUIÇÃO COMPACTA PROTEGIDA – CLASSE 15 kV*. Citado na página 34.

[41] Energisa, *Norma de Distribuição Unificada NDU – 016, "Compatibilização da Arborização com as Redes de Distribuição de Energia Elétrica"ENERGISA/C-GTCD-NRM/Nº170/2018*. Citado 3 vezes nas páginas 34, 36 e 37.

[42] L. C. Bernacci, F. R. Martins, and F. A. M. d. Santos, "Estrutura de estádios ontogenéticos em população nativa da palmeira syagrus romanzoffiana (cham.) glassman (arecaceae)," *Acta Botanica Brasilica*, vol. 22, no. 1, pp. 119–130, 2008. Citado na página 41.

[43] L. Aziz, S. bin Haji Salam, and S. Ayub, "Exploring deep learning-based architecture, strategies, applications and current trends in generic object detection: A comprehensive review," *IEEE Access*, 2020. Citado na página 41.

[44] "Fizyr - deep learning for vision guided robotics. keras implementation of retinanet object detection." Acessado em Março de 2021. Citado 2 vezes nas páginas 41 e 43.

[45] F. Sultana, A. Sufian, and P. Dutta, "Evolution of image segmentation using deep convolutional neural network: a survey," *Knowledge-Based Systems*, vol. 201, p. 106062, 2020. Citado na página 43.

[46] N. S. Keskar and R. Socher, "Improving generalization performance by switching from adam to sgd," *arXiv preprint arXiv:1712.07628*, 2017. Citado na página 45.

[47] ANEEL, "Indicadores coletivos de continuidade." Acessado em Março de 2021. Citado na página 74.

[48] X. Yang, Q. Liu, J. Yan, A. Li, Z. Zhang, and G. Yu, "R3det: Refined single-stage detector with feature refinement for rotating object," *arXiv preprint arXiv:1908.05612*, 2019. Citado na página 74.

[49] V. N. Nguyen, R. Jenssen, and D. Roverso, "Ls-net: Fast single-shot line-segment detector," *arXiv preprint arXiv:1912.09532*, 2019. Citado na página 74.