UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

VICTOR BARPP GOMES

USO DE AMOSTRAS SINTÉTICAS NO TREINAMENTO DE MODELOS PARA A IDENTIFICAÇÃO DO ESTADO DE CHAVES SECCIONADORAS EM SUBESTAÇÕES DE TRANSMISSÃO DE ENERGIA ELÉTRICA

CURITIBA

2024

VICTOR BARPP GOMES

USO DE AMOSTRAS SINTÉTICAS NO TREINAMENTO DE MODELOS PARA A IDENTIFICAÇÃO DO ESTADO DE CHAVES SECCIONADORAS EM SUBESTAÇÕES DE TRANSMISSÃO DE ENERGIA ELÉTRICA

Use of synthetic samples on model training for disconnect switch state identification in transmission substations

Dissertação de Mestrado apresentada como requisito para obtenção do título de Mestre em Computação Aplicada do Programa de Pós-Graduação em Computação Aplicada da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Bogdan Tomoyuki Nassu

CURITIBA

2024

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Curitiba

VICTOR BARPP GOMES

USO DE AMOSTRAS SINTÉTICAS NO TREINAMENTO DE MODELOS PARA A IDENTIFICAÇÃO DO ESTADO DE CHAVES SECCIONADORAS EM SUBESTAÇÕES DE TRANSMISSÃO DE ENERGIA ELÉTRICA

Trabalho de pesquisa de mestrado apresentado como requisito para obtenção do título de Mestre Em Computação Aplicada da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Engenharia De Sistemas Computacionais.

Data de aprovação: 15 de Fevereiro de 2024

Dr. Bogdan Tomoyuki Nassu, Doutorado - Universidade Tecnológica Federal do Paraná

Dr. Helio Pedrini, Doutorado - Universidade Estadual de Campinas (Unicamp)

Dr. Joao Alberto Fabro, Doutorado - Universidade Tecnológica Federal do Paraná

Dedico este trabalho à minha família, pelo
amparo incessante durante estes longos anos
de estudo e pesquisa.

AGRADECIMENTOS

Agradeço ao meu orientador Prof. Dr. Bogdan Tomoyuki Nassu, pelo papel fundamental na formação da metodologia utilizada nesta pesquisa, além do auxílio na revisão desta dissertação. Também agradeço aos professores do Programa de Pós-Graduação em Computação Aplicada (PPGCA) e demais servidores da Universidade Tecnológica Federal do Paraná (UTFPR).

Agradeço a meus familiares, pelo suporte ao longo de toda a minha permanência no programa de mestrado e pelo incentivo a perseverar durante os momentos difíceis.

Agradeço à Copel Geração e Transmissão S.A. (Copel GeT) e ao Instituto de Tecnologia para o Desenvolvimento (Lactec), este último onde trabalho, pela oportunidade de desenvolver esta pesquisa como parte de um projeto de pesquisa e desenvolvimento. Agradeço, ainda, aos colegas de projeto de ambas as empresas, em especial Marcos Scremin, Bruno Marchesi, Lourival Lippmann Junior e Yuri Arnold Gruber, pela dedicação nas atividades que contribuíram diretamente para esta pesquisa — sua colaboração foi fundamental para alcançar estes resultados. Também agradeço ao meu colega de trabalho Ananías Ambrosio Quispe, que, apesar de não fazer parte deste projeto, ajudou-me com dicas e conselhos relacionados à vida acadêmica.

Por fim, agradeço àqueles que porventura não tenha mencionado explicitamente nestas palavras, mas que tenham contribuído de alguma forma, seja em atividades relacionadas ao projeto, atividades do dia-a-dia, apoio material, intelectual, mental, emocional, ou até com ações que possam ter passado despercebidas.

Esta pesquisa foi desenvolvida com recursos do Programa de Pesquisa e Desenvolvimento da Agência Nacional de Energia Elétrica (ANEEL), como parte do projeto de código PD-06491-0455/2017, intitulado “Sistema para determinação automática do estado e temperatura de chaves seccionadoras em subestações”, celebrado entre a Copel GeT e o Lactec.

“Todos os homens, por natureza, tendem ao saber. Sinal disso é o amor pelas sensações. De fato, eles amam as sensações por si mesmas, independentemente da sua utilidade e amam, acima de todas, a sensação da visão. Com efeito, não só em vista da ação, mas mesmo sem ter nenhuma intenção de agir, nós preferimos o ver, em certo sentido, a todas as outras sensações. E o motivo está no fato de que a visão nos proporciona mais conhecimentos do que todas as outras sensações e nos torna manifestas numerosas diferenças entre as coisas.”

(ARISTÓTELES, Metafísica I (Alfa), 980a)

RESUMO

A geração de dados sintéticos está sendo crescentemente explorada na visão computacional devido à capacidade de produzir grandes volumes de dados extremamente variados e rotulados automaticamente, minimizando custos e preenchendo lacunas existentes em conjuntos de dados reais. Dados sintéticos são especialmente interessantes para viabilizar aplicações que impõem dificuldades na coleta de dados reais em quantidade e variedade suficientes para o aprendizado de máquina. Contudo, os dados são gerados a partir de simplificações da realidade, como simulações computacionais e modelos estatísticos, levando a uma distanciação entre o domínio sintético e o real e, consequentemente, a uma dificuldade na transferência de aprendizado. Este trabalho apresenta uma avaliação experimental de dados sintéticos em uma aplicação particular: o reconhecimento do estado de chaves seccionadoras em subestações de transmissão, um problema de grande importância para o setor elétrico, mas que está sujeito às dificuldades de um ambiente repleto de restrições de segurança e de equipamentos críticos para o fornecimento de energia. Os dados foram gerados por renderização de modelos tridimensionais de subestações usando randomização de domínio, isto é, a manipulação aleatória dos parâmetros de simulação, incluindo a abertura e fechamento das chaves. A tarefa de reconhecimento foi modelada como um problema de classificação, para o qual foram avaliadas seis arquiteturas de redes convolucionais distintas, treinadas inicialmente apenas com dados sintéticos e, posteriormente, com adição de uma pequena quantidade de dados reais. Em testes com imagens capturadas por câmeras físicas instaladas nas subestações, a maioria dos modelos treinados apenas com dados sintéticos obteve taxas de acerto entre 87% e 97%, demonstrando haver um distanciamento entre os domínios sintético e real que limita o desempenho. Isso foi mitigado com a adição de dados reais no treinamento, com os quais se alcançaram resultados entre 95% e 99%. Ainda, a abordagem proposta se mostrou especialmente eficaz no reconhecimento de eventos raros, com taxas de acerto superiores a 90%.

Palavras-chave: dados sintéticos; randomização de domínio; classificação de imagens; redes neurais convolucionais; subestações.

ABSTRACT

Synthetic data generation is being increasingly explored in computer vision due to its capacity to produce large volumes of highly varied and automatically labeled data. This type of data minimizes costs and can fill gaps in real datasets. Synthetic data are especially relevant to enable applications that impose constraints for collecting real data in sufficient quantity and variety for machine learning. However, the data are generated from simplifications of reality, such as computer simulations and statistical models, creating a gap between the real and synthetic domains, consequently leading to difficulties in transferring the learned features. This work presents an experimental evaluation of synthetic data in a particular application: disconnect switch state recognition in transmission substations, a problem of great importance to the electric sector, but which is subject to the difficulties of a restricted environment full of equipment critical to power delivery. The data were generated by rendering virtual tridimensional models of substations and employing domain randomization, that is, the random manipulation of simulation parameters, including the act of opening and closing switches. We modeled the recognition task as a classification problem and evaluated six distinct convolutional network architectures, initially trained only on synthetic data and later with the addition of a small amount of real data. In tests with images captured by physical cameras installed in substations, most of the models trained solely on synthetic data achieved between 87% and 97% accuracy, demonstrating the existence of a synthetic-to-real domain gap that limits performance. Adding real data to the training set mitigated this effect, raising the accuracy to values between 95% and 99%. Furthermore, the proposed approach performed remarkably well in the recognition of rare events, achieving rates above 90%.

Keywords: synthetic data; domain randomization; image classification; convolutional neural networks; substations.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

Siglas

2D	Bidimensional
3D	Tridimensional
ADR	*Active Domain Randomization*
AP	*Average Precision*
API	*Application Programming Interface*
BTA	Bateias
CCO	Campo Comprido
CNN	*Convolutional Neural Network*
CPU	*Central Processing Unit*
DA	*Domain Adaptation*
DR	*Domain Randomization*
FPN	*Feature Pyramid Network*
GDPR	*General Data Protection Regulation*
GPU	*Graphics Processing Unit*
HTTP	*Hypertext Transfer Protocol*
IEEE	*Institute of Electrical and Electronics Engineers*
ILSVRC	*ImageNet Large-Scale Visual Recognition Challenge*
IoU	*Intersection over Union*
JCR	*Journal Citation Reports*
LDA	*Linear Discriminant Analysis*
LGPD	Lei Geral de Proteção de Dados Pessoais
mAP	*Mean Average Precision*

MDP	*Markov Decision Process*
mIoU	*Mean Intersection over Union*
MMD	*Maximum Mean Discrepancy*
NME	*Normalized Mean Error*
ONS	Operador Nacional do Sistema Elétrico
PNG	*Portable Network Graphics*
PPGCA	Programa de Pós-Graduação em Computação Aplicada
PTZ	*Pan, Tilt, Zoom*
R-CNN	*Region-based Convolutional Neural Network*
R-FCN	*Region-based Fully Convolutional Network*
RGB	*Red, Green, Blue*
S.A.	Sociedade Anônima
SDR	*Structured Domain Randomization*
SE	Subestação
SNPTEE	Seminário Nacional de Produção e Transmissão de Energia Elétrica
SSD	*Single-Shot Detector*
SVM	*Support Vector Machine*
UTFPR	Universidade Tecnológica Federal do Paraná
VOC	*Visual Object Classes*

Acrônimos

ANEEL	Agência Nacional de Energia Elétrica
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
COCO	*Common Objects in Context*
FLOPS	*Floating-Point Operations Per Second*
GAN	*Generative Adversarial Network*
GAP	*Global Average Pooling*
HOG	*Histogram of Oriented Gradients*

LiDAR	*Light Detection And Ranging*
MOTA	*Multi-Object Tracking Accuracy*
PASCAL	*Pattern Analysis, Statistical Modelling and Computational Learning*
ReLU	*Rectified Linear Unit*
ResNet	*Residual Neural Network*
SCADA	*Supervisory Control And Data Acquisition*
SIFT	*Scale-Invariant Feature Transform*

SUMÁRIO

1 INTRODUÇÃO

Conforme explicado por McDonald (2017), chaves seccionadoras são equipamentos mecânicos que conduzem corrente elétrica e servem como um ponto de abertura em um circuito para isolamento de um equipamento de subestação, tal como um disjuntor ou transformador. Tipicamente, uma chave seccionadora é instalada em cada lado do equipamento, permitindo que a equipe de manutenção tenha uma confirmação visual de que o circuito está aberto e que o equipamento está seguro para o trabalho. Também é comum a instalação de chaves seccionadoras com a função de *bypass* em alguns equipamentos, as quais permanecem normalmente abertas, mas podem ser fechadas para desviar a corrente elétrica por um caminho alternativo. Isso é normalmente aliado ao redirecionamento de circuitos para um barramento de transferência, permitindo a realização de manutenção em disjuntores sem a interrupção no fornecimento de energia.

Há diversos modelos de chaves seccionadoras, com variadas construções mecânicas e padrões de movimentação, cada qual aplicável em diferentes cenários e níveis de tensão. Pode-se classificá-las conforme a forma de abertura, sendo que alguns dos principais tipos são: dupla abertura, abertura central e semi-pantográfica. A Figura 1 mostra esses tipos de chaves.

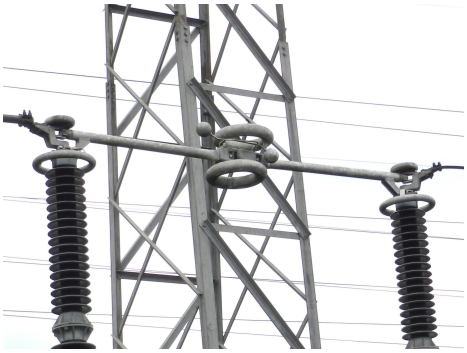
A abertura ou fechamento de uma chave seccionadora, dependendo do modelo, pode ser feita de forma manual, por emprego de força física de uma pessoa sobre uma alavanca ou outro mecanismo, ou de forma automática, por meio de motores e controles eletrônicos. Essas últimas são frequentemente interligadas a sistemas de supervisão e aquisição de dados (SCADA), possibilitando o controle remoto das chaves a partir do centro de operação da concessionária de energia. Como os circuitos de uma subestação são trifásicos na grande maioria das situações, as chaves seccionadoras são instaladas em conjuntos de três, uma para cada fase do circuito. Normalmente, as três chaves são acopladas mecanicamente, sendo atuadas em conjunto durante a abertura ou fechamento, porém existem modelos que permitem manobra de cada fase individualmente.

A determinação da situação de chaves seccionadoras de subestações é uma necessidade recorrente das concessionárias de transmissão de energia elétrica. Ao executar manobras nas chaves, seja por motivos operacionais ou de manutenção, há a possibilidade de ocorrência de falhas de operação, já que, apesar de as chaves serem geralmente motorizadas e controláveis remotamente, podem ocorrer problemas mecânicos na movimentação dos equipamentos. Uma chave seccionadora está sujeita às intempéries do meio-ambiente e pode, em alguns casos, permanecer durante anos em um mesmo estado (aberto ou fechado), levando à degradação gradual de seus mecanismos de movimentação. Consequentemente, após um procedimento desse tipo, é possível que uma chave não tenha sido efetivamente manobrada, ou a manobra tenha sido executada incorretamente. Como resultado, é possível que uma chave esteja aberta quando deveria estar fechada, ou vice-versa.

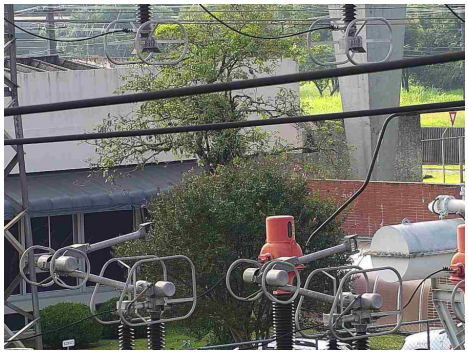
(a) Dupla abertura, aberta

(b) Dupla abertura, fechada

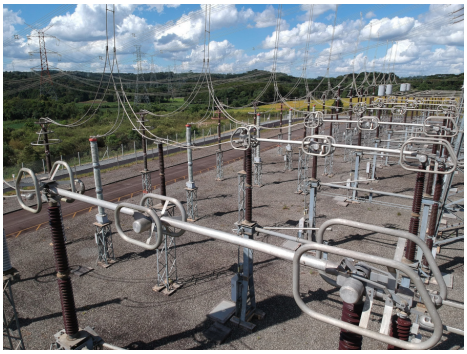
(c) Abertura central, aberta



(d) Abertura central, fechada



(e) Semi-pantográfica vertical, aberta



(f) Semi-pantográfica horizontal, fechada

Figura 1 – Exemplos de chaves seccionadoras de tipos diversos.
Fonte: Autoria própria (2024).

O restabelecimento dos circuitos afetados por um procedimento de manobra em chaves seccionadoras apenas pode ser feito após a conferência de que as chaves estão no estado correto, pois em caso contrário, há risco de sérios danos aos equipamentos da subestação, resultando em prejuízo para a empresa, além de desligamentos não previstos. Geralmente, a tarefa de conferir os estados das chaves seccionadoras é executada por um profissional da concessionária, que deve se deslocar até o local da subestação e fazer uma inspeção visual. Todavia, há a probabilidade de que a pessoa responsável pela conferência cometa um erro de julgamento devido à fadiga, falta de atenção, inexperiência ou outros motivos. Além do custo do deslocamento e do risco de erro humano, o tempo de duração do desligamento provoca

descontos na receita da concessionária, conforme as normas de Parcela Variável da Agência Nacional de Energia Elétrica (ANEEL, 2020).

1.1 Motivação

Considerando a importância da informação do estado das chaves seccionadoras, é de grande interesse das concessionárias um sistema capaz de fazer o reconhecimento de forma autônoma. Entre as alternativas propostas para esta tarefa, há aquelas baseadas em sensores instalados individualmente por chave, como os trabalhos de Semedo, Oliveira e Cardoso (2014) e Werneck e Allil (2019), as quais têm a vantagem de medir uma grandeza física que indica o estado do equipamento com grande precisão. Porém, essas soluções possuem a desvantagem da complexidade da instalação e manutenção, devido aos altos níveis de tensão, além do custo associado à adição de equipamentos a cada uma das chaves. Por isso, buscou-se, neste projeto, uma solução que operasse totalmente desvinculada dos equipamentos da subestação, sem exigir modificações nos circuitos. A alternativa encontrada foi o desenvolvimento de um sistema de reconhecimento do estado das chaves por imagens capturadas por câmeras instaladas em pontos fixos da subestação, utilizando como informação apenas o aspecto original dos equipamentos (ou seja, sem pintar as chaves ou adicionar adesivos para facilitar a detecção, o que tornaria o sistema invasivo).

Tendo definido o uso de câmeras para essa tarefa, ainda é necessário encontrar a técnica de processamento de imagem mais adequada. À primeira vista, a geometria relativamente retilínea das chaves parece sugerir que uma estratégia simples baseada em detecção de bordas e/ou retas, tal como a transformada de Hough (BALLARD, 1981), teria bons resultados. Entretanto, uma subestação é um ambiente visualmente ambíguo, com outras estruturas como torres, pórticos e cabos, na mesma orientação das chaves, aparecendo ao fundo na mesma região da imagem que uma chave. Além disso, por se tratar de um local aberto, está sujeito a variações de ambiente, como a entrada ou saída de um objeto estranho, mudanças de iluminação, entre outros fatores. A isso se soma a grande variedade de aspectos que as chaves podem ter, assim como a variedade de ângulos de visão sob os quais uma chave pode ser monitorada — isto dificulta a criação de soluções que funcionem de maneira geral sem a necessidade de ajustes às condições específicas de cada chave individual. Para que o sistema seja capaz de lidar com essas características, considerou-se necessária a aplicação de uma técnica baseada em aprendizado de máquina.

Um trabalho anterior com uso de visão computacional foi realizado por Nassu *et al.* (2022) em subestações de distribuição, as quais possuem chaves seccionadoras do tipo “faca”, cuja geometria é mais simples em relação às chaves usadas na transmissão. Devido a isso, e por conta da variedade de tipos de chaves de transmissão, não é possível incorporar muitas suposições a respeito da forma e do movimento das chaves na modelagem das soluções, necessitando de um método mais genérico do que os modelos propostos naquele trabalho. En-

tretanto, algumas questões levantadas também se aplicam aqui. Como as chaves raramente são manobradas, as imagens apresentadas ao algoritmo de treinamento sempre contêm as mesmas chaves nos mesmos estados, fazendo com que os modelos possam aprender a tomar decisões com base em objetos de fundo ou outras características estranhas às chaves seccionadoras (NASSU *et al.*, 2022). Idealmente, para ter um treinamento mais robusto, todas as chaves deveriam ser manobradas e mantidas no estado oposto por vários dias e noites para a coleta de imagens, visando compor um conjunto de dados (*dataset*) de treinamento em que cada chave aparecesse tanto no estado aberto quanto no estado fechado. Isso é operacionalmente inviável devido a questões elétricas, pois não se pode interromper arbitrariamente o fornecimento de energia aos consumidores. A execução de manobras em chaves seccionadoras, mesmo que por curta duração (por exemplo, um dia), exige fortes justificativas, além de autorizações da administração da concessionária e até do ONS.

Essa dificuldade na obtenção de dados suficientemente representativos do universo de possíveis posições de chaves é a principal motivação desta pesquisa. A abordagem proposta neste trabalho visa contornar esse problema com a geração de dados sintéticos, o que já se mostrou eficaz em diversas aplicações (NIKOLENKO, 2021). No presente projeto, buscou-se desenvolver um *framework* de geração de imagens por renderização de modelos tridimensionais paramétricos da subestação, os quais permitem controlar não apenas a posição das chaves, mas também o enquadramento de imagem (posição da câmera), propriedades de iluminação e outras características. Com essas ferramentas, torna-se possível construir um conjunto de dados contendo imagens das chaves em posições diferentes daquelas observadas na situação rotineira da subestação, sem qualquer impacto no sistema elétrico. As imagens geradas foram utilizadas no aprendizado de modelos profundos para classificação do estado de chaves (aberto ou fechado) e o desempenho foi testado com imagens reais, capturadas por câmeras físicas instaladas nas subestações.

Outra motivação do projeto é a implantação do sistema de reconhecimento em um computador instalado no local da subestação. Isso se deve ao fato de que subestações de alta tensão estão normalmente afastadas dos grandes centros e, por consequência, enfrentam dificuldades de telecomunicação: raramente há disponibilidade de fibra ótica, e em muitos casos também não há cobertura de rede celular. As opções viáveis, em geral, são conexões via satélite ou a própria rede de automação da concessionária, as quais não comportam o tráfego constante de imagens de alta resolução. Assim, fica inviabilizado o processamento remoto das imagens, seja no centro de operação da concessionária ou em plataformas de computação em nuvem. Além disso, o computador utilizado para a execução do reconhecimento deve preferencialmente ser uma plataforma embarcada ao invés de um computador padrão de alto desempenho, devido à limitação de espaço físico na subestação e à necessidade de manter operação contínua sem intervenção humana. O equipamento deve ser, inclusive, resistente a variações de tensão e capaz de restabelecer seu funcionamento automaticamente após uma falta de energia. Essas

restrições impactam os modelos de inteligência artificial, que precisam ser dimensionados para execução em um computador com especificações limitadas.

1.2 Objetivos

Tendo em vista o contexto da dificuldade de se obter dados representativos de chaves seccionadoras em ambos os estados, esta pesquisa busca responder a uma pergunta central: é possível resolver o problema de reconhecimento do estado de chaves seccionadoras com redes neurais convolucionais (CNN) treinadas em imagens sintéticas geradas a partir de modelos 3D?

Para responder a essa pergunta, traçou-se como objetivo geral avaliar experimentalmente o uso de dados sintéticos no treinamento de redes neurais convolucionais para o reconhecimento do estado de chaves seccionadoras em subestações de transmissão. Além de avaliar o desempenho dos modelos de forma geral, os experimentos visam verificar a capacidade dos modelos em detectar mudanças de estado. Esse objetivo também inclui verificar se há um limite no desempenho em função da quantidade de amostras sintéticas fornecidas no treinamento.

Para atingir o objetivo geral, destacam-se diversos objetivos específicos. Foi necessário realizar uma revisão bibliográfica sobre as técnicas normalmente utilizadas para a criação ou ampliação de conjuntos de treinamento para aprendizado de máquinas em visão computacional, identificando aquelas que poderiam ser exploradas no cenário proposto. Também se mostrou necessária a coleta de imagens em situações reais de operação de subestações de transmissão, tanto para a análise de suas propriedades quanto para a realização de experimentos posteriores. Para o treinamento de modelos computacionais, foi necessária a geração de imagens a partir de modelos tridimensionais de subestações, incluindo a parametrização e considerações sobre as propriedades que poderiam permitir a transferência do aprendizado realizado sobre estes dados sintéticos para aplicação sobre imagens reais. Também se destacam como objetivos a produção de conjuntos de dados com base nas imagens coletadas e produzidas; o desenvolvimento de protótipos que implementassem o treinamento e teste de redes convolucionais para a identificação do estado das chaves, de forma a permitir a avaliação experimental sistemática das técnicas consideradas; e a implementação de uma versão piloto da solução proposta que pudesse ser executada sobre uma plataforma embarcada.

Além disso, como esta pesquisa está inserida no contexto de um projeto de pesquisa e desenvolvimento do setor elétrico, há, também, um objetivo ligado à inovação: implementar uma versão piloto da solução proposta que pudesse ser utilizada na prática em subestações de transmissão. Esse objetivo estabelece uma série de requisitos, tais como a restrição de instalação de câmeras em estruturas pré-existentes das subestações e a execução do reconhecimento em sistemas embarcados localmente instalados, devido às restrições de telecomunicação. Várias decisões tomadas ao longo da concepção da solução foram guiadas por essa demanda, principalmente na opção por alternativas com maior eficiência computacional.

1.3 Abordagem proposta

Para a abordagem proposta neste trabalho, foram instaladas câmeras em duas subestações no estado do Paraná, com o objetivo de coletar dados para teste dos modelos de visão computacional. As subestações foram mapeadas e reconstruídas virtualmente na forma de modelos 3D, sobre os quais se desenvolveu um *framework* de geração de dados por renderização. Para a tarefa de visão computacional, foram elaboradas três arquiteturas de redes convolucionais distintas e selecionadas outras três arquiteturas amplamente utilizadas em aplicações gerais.

O treinamento dos modelos foi realizado separadamente para cada tipo de chave da subestação, em um primeiro momento utilizando puramente dados sintéticos, e em um segundo momento acrescentando uma pequena quantidade de dados reais. Na etapa de teste, foram utilizadas capturas reais das câmeras instaladas nas subestações, visando demonstrar a capacidade de se utilizar o aprendizado no domínio sintético para inferência no domínio real. Os dados utilizados nos processos de treinamento totalizam cerca de 1,15 milhão de amostras sintéticas e 6 mil amostras reais (entre os diferentes tipos de chave), enquanto o conjunto de teste é formado por aproximadamente 62 mil amostras reais.

Nos experimentos descritos no presente trabalho, observou-se que, de forma geral, a transferência de aprendizado a partir de um domínio puramente sintético foi bem sucedida, embora com limitações, alcançando taxas de acerto entre 87% e 97% na maioria dos modelos. Ao adicionar a parcela de dados reais no treinamento, os resultados, em sua maioria, passaram a figurar entre 95% e 99%, a depender do tipo de chave.

As principais contribuições deste trabalho incluem:

- a proposta de uma solução econômica para o monitoramento de chaves seccionadoras em subestações de transmissão, utilizando câmeras comercialmente acessíveis e uma plataforma de processamento embarcada de baixo custo;
- a estratégia de lidar com a dificuldade de se obter um conjunto de dados representativo por meio da geração de dados sintéticos a partir de modelos tridimensionais virtuais de subestações;
- a influência da adição de uma pequena quantidade de amostras reais às imagens sintéticas, aproximando os dois domínios;
- a comparação de desempenho entre diversos modelos baseados em redes neurais convolucionais para a tarefa de classificação do estado de chaves;
- a demonstração experimental de que a solução proposta consegue alcançar resultados superiores aos dos trabalhos anteriores no reconhecimento do estado de chaves, lidando inclusive com tipos diferentes de chave, pontos de vista variados, entre outros desafios.

Este trabalho foi desenvolvido como parte do projeto PD-06491-0455/2017 do Programa de Pesquisa e Desenvolvimento Tecnológico da ANEEL, intitulado “Sistema para determinação automática do estado e temperatura de chaves seccionadoras em subestações”, contratado pela Copel Geração e Transmissão S.A. e executado pelo Instituto de Tecnologia para o Desenvolvimento (Lactec). Houve contribuição de terceiros em partes do trabalho, especialmente na instalação de câmeras, coleta de imagens, mapeamento de subestações e construção de modelos 3D, as quais são devidamente indicadas nas respectivas seções.

Durante o desenvolvimento do projeto, foi publicado o artigo Nassu *et al.* (2022), intitulado “*A Computer Vision System for Monitoring Disconnect Switches in Distribution Substations*”, no periódico “*IEEE Transactions on Power Delivery*”, que possui fator de impacto 4,825 (JCR 2022) e Qualis A1 (avaliação CAPES 2017–2020). Também foi publicado o artigo Lippmann Jr *et al.* (2022), intitulado “Ferramenta de escolha de pontos ótimos para posicionamento de câmeras para observação de elementos estratégicos dos circuitos das subestações de transmissão”, no XXVI Seminário Nacional de Produção e Transmissão de Energia Elétrica (SNPTEE), considerado o maior evento do setor elétrico brasileiro. Ademais, o artigo Gomes *et al.* (2024), contendo resultados desta dissertação, intitulado “*Exploring Synthetic Data for Training Deep Learning Models for High-Voltage Disconnector State Identification*”, foi submetido ao periódico “*IEEE Transactions on Power Delivery*” e está em revisão. Os três artigos tiveram contribuição direta e significativa do autor do presente trabalho.

1.4 Estrutura do trabalho

Este trabalho está organizado da seguinte maneira: o Capítulo 2 introduz técnicas aplicadas na geração e utilização de dados sintéticos em visão computacional e apresenta algumas arquiteturas de redes convolucionais populares. O Capítulo 3 apresenta um levantamento de trabalhos anteriores que utilizaram dados sintéticos para aprendizado de máquina. O Capítulo 4 descreve os materiais, técnicas e procedimentos utilizados na abordagem proposta. O Capítulo 5 discute os experimentos e resultados obtidos. Por fim, o Capítulo 6 apresenta as conclusões obtidas e trabalhos futuros. Como complemento, o Apêndice A reapresenta os resultados detalhadamente em forma de tabelas.

2 REFERENCIAL TEÓRICO

Segundo Andrew Ng, professor da Universidade de Stanford, co-fundador da plataforma de cursos online Coursera e do time de pesquisa Google Brain, renomado por sua pesquisa em aprendizado de máquina e inteligência artificial, em entrevista a Strickland (2022), o paradigma dominante em inteligência artificial na última década foi manter o conjunto de dados fixo e focar em aprimorar o código. Seguindo este princípio, diversos algoritmos e arquiteturas, voltados especialmente para *deep learning*, foram desenvolvidos, obtendo grandes avanços sobre conjuntos de dados fixos em desafios e *benchmarks*, tais como ImageNet (DENG *et al.*, 2009), PASCAL VOC (EVERINGHAM *et al.*, 2010) e COCO (LIN *et al.*, 2014). Estes avanços foram tais que, para muitas aplicações práticas, pode-se dizer que a arquitetura é um problema resolvido. Por causa disso, nesses casos práticos tornou-se mais produtivo fixar a arquitetura do modelo e focar, ao invés disso, em aprimorar o conjunto de dados e a forma como os dados disponíveis são explorados durante o treinamento. Isso consolida a inteligência artificial como uma disciplina centrada em dados.

Contudo, em problemas encontrados na prática, nem sempre é possível obter um conjunto de dados com a representatividade e variedade necessárias. Isso pode ocorrer por diversos motivos:

1. Diversidade limitada. Situações raras ou difíceis de capturar no mundo real podem estar sub-representadas em um conjunto de dados, levando ao desbalanceamento entre classes. Alguns domínios também estão expostos a condições de grande variabilidade, sendo inviável mapear e coletar dados sob uma quantidade satisfatória de condições.

2. Privacidade e regulações. Informações pessoais estão sujeitas a leis e regulamentações governamentais, tais como a *General Data Protection Regulation* (GDPR) na Europa e a Lei Geral de Proteção de Dados Pessoais (LGPD) no Brasil. As restrições na coleta e uso de dados pessoais dificultam a formação de um *dataset* para tarefas de inteligência artificial.

3. Custo e tempo para coleta de dados. Dependendo do domínio do problema, a própria coleta de dados no mundo real pode ser uma tarefa custosa (por exemplo, necessitando de equipamentos especiais).

Mesmo nos casos em que existem bases de dados extensas, para muitos projetos também é necessário que as amostras estejam rotuladas, ou seja, além do dado bruto, precisa-se de informações adicionais sobre o seu significado. Usualmente, isso é feito por seres humanos, os quais observam cada uma das amostras e dão seu julgamento. Em imagens, por exemplo, isso pode envolver classificação de amostras, marcação de pontos de interesse, marcação de caixas delimitadoras ao redor de objetos específicos, segmentação pixel-a-pixel, entre outras

atividades. Assim, a formação de um conjunto de dados pode se tornar bastante lenta e dispendiosa, especialmente se o trabalho de rotulagem precisar ser realizado por peritos.

O presente trabalho está sujeito a essas limitações. Como subestações são, em geral, ambientes externos, estão expostas a variações de luz solar durante o dia e também ao longo do ano (devido à latitude), além de eventos meteorológicos como chuva e neblina, sendo necessário coletar dados em diversos momentos para obter a melhor diversidade possível. Porém, subestações também são ambientes de risco elevado, tendo acesso restrito apenas a pessoas autorizadas para realizar determinada atividade, além da exigência de cursos e treinamentos específicos. Ou seja, não é possível obter dados suficientes para o aprendizado de máquina simplesmente com pessoas fotografando os objetos a partir do chão com câmeras portáteis em visitas ocasionais. Adicionalmente, o posicionamento de câmeras para o monitoramento exige um planejamento prévio e a execução de obras, incluindo projeto, autorizações junto à concessionária, negociações com empresas qualificadas para a execução da obra, compra de câmeras e outros materiais (como cabos de alimentação e de comunicação com blindagem eletromagnética), entre outras necessidades, elevando o custo de preparação da infraestrutura de coleta de dados. Mesmo após todos esses esforços, há uma dificuldade ainda mais severa para o problema do reconhecimento do estado de chaves seccionadoras: não é possível realizar uma grande quantidade de manobras, sob condições variadas de iluminação e clima, sem perturbar o funcionamento normal da subestação.

Limitações semelhantes às enfrentadas neste trabalho também aparecem em outros domínios, levando a comunidade científica a procurar alternativas, entre as quais surgiu a geração de dados por computador, denominados dados sintéticos (ANDREWS, 2021). Além de possibilitar a geração de uma quantidade arbitrária de dados, o computador pode também emitir as informações de rotulagem para cada amostra de forma automática, e em certos casos até emitir informações adicionais que um rotulador humano não seria capaz de fornecer. Outra vantagem dos dados sintéticos é a capacidade de incluir exemplos de situações raras ou dificilmente observáveis no domínio real, mas cruciais para garantir o correto funcionamento de um sistema. Dados sintéticos já foram aplicados com sucesso em problemas como detecção de objetos (BORKMAN *et al.*, 2021; HINTERSTOISSER *et al.*, 2019), detecção de veículos (GAIDON *et al.*, 2016; TREMBLAY *et al.*, 2018; PRAKASH *et al.*, 2019), segmentação semântica (SANKARANARAYANAN *et al.*, 2018; CHEN *et al.*, 2019), controle robótico (TOBIN *et al.*, 2017; JAMES *et al.*, 2019), detecção de pessoas (EBADI *et al.*, 2021; TRIPATHI *et al.*, 2019) e análise de rosto (WOOD *et al.*, 2021).

Dados sintéticos podem ser criados por simulações computacionais e modelos estatísticos. Em projetos de visão computacional, a geração de dados sintéticos é feita principalmente com simulações, usando técnicas de computação gráfica tais como modelagem tridimensional, renderização e composição. Também são utilizados modelos generativos estatísticos, como *variational autoencoders* e redes adversárias generativas (GANs), que geram dados derivados de um processo de aprendizado sobre dados reais.

Apesar das vantagens mencionadas acima, na maior parte das situações, um conjunto de dados sintético não consegue exprimir fielmente as características do mundo real, pois é gerado a partir de aproximações da realidade. Cada técnica possui um viés associado ao próprio processo de geração. No caso de uma simulação com modelagem 3D, por exemplo, as imagens renderizadas normalmente têm poucas imperfeições em formato e textura, diferentemente da situação real. Essa distanciação entre o domínio sintético e real (*domain gap*) é o principal desafio em sua utilização para inteligência artificial. Algumas estratégias usadas para lidar com isso incluem melhorar o realismo do gerador de dados, utilizar técnicas de adaptação de domínio no modelo ou então randomizar uma grande quantidade de parâmetros do gerador.

2.1 Técnicas de geração de dados sintéticos e transferência de aprendizado

Nesta seção, são descritas algumas técnicas comuns a trabalhos que fazem uso de imagens sintéticas para visão computacional.

2.1.1 Modelagem 3D

Na área de computação gráfica, um modelo geométrico é uma representação de algo que se deseja mostrar em uma imagem (HUGHES *et al.*, 2013). Modelos tridimensionais (3D) são formados por malhas de polígonos (em geral triângulos) descritos por vértices posicionados no espaço cartesiano $\mathbb{R}^3$. Além da informação geométrica, um modelo possui atributos adicionais que descrevem os materiais que compõem o objeto, como cor, textura e refletividade, cuja função é determinar como o objeto interage com a luz.

Instâncias de um modelo podem ser posicionadas em uma cena (que também é um modelo) e modificadas por meio de transformações geométricas, como translação, rotação e escala, para compor o ambiente desejado. Uma cena também contém fontes de luz (que possuem atributos como cor, direção e intensidade) e observadores, frequentemente chamados de câmeras.

O ato de se criar um modelo é chamado de modelagem. Usualmente, isso é feito de forma manual, com o uso de *software* especializado (por exemplo, 3ds Max[1], Maya[2] ou Blender[3]). Trata-se de um trabalho similar a artes plásticas como a escultura (YU *et al.*, 2011). Para auxiliar o artista no processo de modelagem de objetos existentes no mundo real, é frequente a utilização de informações de referência, como medidas, desenhos técnicos, fotografias e até escaneamentos (obtidos com tecnologias como LiDAR[4]).

[1] https://www.autodesk.com.br/products/3ds-max/overview
[2] https://www.autodesk.com.br/products/maya/overview
[3] https://www.blender.org/
[4] Acrônimo para *light detection and ranging* (SHAN; TOTH, 2018)

Também é possível gerar modelos 3D de forma automática. A técnica de fotogrametria, por exemplo, possibilita estimar as coordenadas tridimensionais de superfícies a partir de um conjunto de fotografias de pontos de vista distintos, como demonstrado por Oliveira (2002), Gruen, Remondino e Zhang (2004) e Koutsoudis, Arnaoutoglou e Chamzas (2007). Outras técnicas de modelagem automática utilizam como entrada nuvens de pontos capturadas por escaneamento, tais como os trabalhos de Kolluri, Shewchuk e O'Brien (2004), Huang *et al.* (2009) e Berger *et al.* (2014). Há também a possibilidade de geração procedural de modelos 3D, que pode ser feita nos casos em que a geometria do objeto pode ser descrita facilmente por um conjunto de regras e/ou funções matemáticas. Alguns exemplos podem ser encontrados em Fowler, Meinhardt e Prusinkiewicz (1992) e Whiting, Ochsendorf e Durand (2009).

O processo de conversão de uma descrição de uma cena tridimensional em uma imagem é chamado de renderização (PHARR; JAKOB; HUMPHREYS, 2016). Quase todos os sistemas de renderização fotorrealista usam como base o algoritmo de *ray-tracing*, cujo funcionamento se baseia em seguir o caminho de um raio de luz através da cena, o qual interage com os objetos do ambiente. Por outro lado, sistemas de renderização em tempo real frequentemente utilizam técnicas de rasterização, com custo computacional inferior, mas com menor precisão física.

Cenas 3D podem ser manipuladas de forma procedural para produzir um conjunto de dados sintético por meio de renderização. Objetos individuais podem ter sua posição, rotação e tamanho (escala) alterados, bem como as propriedades de seus materiais (por exemplo, cor, textura, refletividade e transparência). A malha de triângulos do objeto pode ser modificada, causando alterações de forma. Objetos adicionais podem ser posicionados na cena, podendo causar oclusão. A iluminação da cena pode ser modificada, podendo inclusive gerar sombra sobre os objetos. A posição e rotação da câmera permitem observar a cena a partir de qualquer ponto de vista. Essas e outras propriedades permitem que as imagens geradas a partir de um modelo 3D tenham grande diversidade de características. Por outro lado, a construção de modelos 3D é trabalhosa, podendo demandar a atuação de especialistas, principalmente quando é necessário um grau de fidelidade elevado. Em algumas situações, uma alternativa pode ser o uso de modelos já existentes, mas quanto maior a especificidade da aplicação, menor a chance de que um modelo pré-existente atenda às necessidades. Por isso, antes de considerar essa técnica para a geração de dados sintéticos, deve-se ponderar se os benefícios da modelagem 3D superam os empecilhos para se coletar dados reais.

2.1.2 Composição 2D

A combinação de elementos de diversas imagens para gerar uma nova imagem é chamada de composição (PORTER; DUFF, 1984). Essa técnica é tradicionalmente usada no cinema: os atores são gravados em uma sala com paredes pintadas de uma cor específica (frequentemente verde). Depois, a gravação é processada, substituindo os *pixels* verdes por outra

imagem que contém o plano de fundo desejado. A substituição não é binária — *pixels* parcialmente verdes são substituídos por uma combinação entre a gravação dos atores e o plano de fundo (HUGHES *et al.*, 2013).

Imagens digitais coloridas são normalmente descritas na forma de três matrizes 2D (canais), cada uma representando a intensidade de cor de cada pixel no modelo RGB: vermelho (R — *red*), verde (G — *green*) e azul (B — *blue*). Para facilitar a composição, pode-se incluir um canal adicional chamado alfa (α), que simboliza a opacidade de cada *pixel*. Considerando a escala do canal alfa entre 0 e 1, o valor $\alpha = 0$ significa que o *pixel* é totalmente transparente, $\alpha = 1$ representa opacidade total e frações representam transparência parcial. Na composição de imagens, o canal alfa controla a mistura de cores entre a imagem de primeiro plano e a imagem de fundo.

O canal alfa pode ser introduzido em uma imagem existente, seja por uma técnica de *chroma keying* (como a substituição do fundo verde), edição direta em um *software* especializado ou outras maneiras. Em computação gráfica, o algoritmo de renderização pode gerar a informação do canal alfa automaticamente, pois a geometria dos objetos é conhecida (HUGHES *et al.*, 2013).

Para a geração de dados sintéticos, a composição 2D é utilizada principalmente para adicionar planos de fundo a imagens renderizadas. Outra abordagem é fazer composição com elementos de imagens reais ao invés de renderizações de modelos 3D (Figura 2), o que pode-se dizer que está na fronteira entre dados sintéticos e *data augmentation* (NIKOLENKO, 2021).

Figura 2 – Imagens sintéticas geradas via composição 2D usando recortes de imagens reais. Fonte: Adaptado de Tripathi *et al.* (2019).

Quanto à variedade dos dados produzidos, como as composições são geradas a partir de um conjunto de imagens, o resultado é restrito pelas imagens originais, o que torna essa técnica útil mas limitada, especialmente quando utilizada de forma isolada. Por outro lado, adicionar variações ao fundo pode ajudar modelos de aprendizado de máquina a isolar os elementos que realmente devem ser considerados para a tarefa, por exemplo, evitando que estruturas visíveis ao fundo interfiram no reconhecimento de um objeto.

2.1.3 Adaptação de domínio

A adaptação de domínio (DA — *domain adaptation*) é uma categoria de técnicas que têm por objetivo fazer um modelo treinado em um domínio de dados, o domínio de origem, funcionar bem em um domínio diferente (domínio alvo). Nas aplicações que utilizam dados sintéticos, em geral o domínio de origem é o sintético e o domínio alvo é o real. Métodos de adaptação de domínio utilizam dados do domínio alvo para fazer algum ajuste estatístico, e podem operar tanto no nível dos dados (buscando aproximar os dois domínios, por exemplo, fazendo os dados sintéticos parecerem mais realistas, como na Figura 3) quanto no nível do modelo (modificando o espaço de *features* ou o processo de treinamento) (NIKOLENKO, 2021). Existe uma grande variedade de métodos, incluindo aprendizado supervisionado (que requer dados rotulados), não supervisionado (que não utiliza dados rotulados) e semi-supervisionado (que utiliza ambos, com uma fração menor de dados rotulados).

Figura 3 – Transformação de imagens sintéticas (esquerda) para torná-las mais realistas (direita) usando adaptação de domínio.

Fonte: Adaptado de Chen *et al.* (2019).

Wang e Deng (2018) sumarizam algumas abordagens de *deep* DA, classificando-as em três principais grupos:

- Baseadas em discrepância: buscam reduzir o desvio entre as distribuições dos dois domínios por meio de ajuste fino, usando algum critério de distância (como a máxima discrepância média, MMD — *maximum mean discrepancy* (GRETTON *et al.*, 2012)).
- Baseadas em aprendizado adversário: um discriminador é treinado para classificar se uma amostra pertence ao domínio de origem ou alvo, enquanto outro modelo busca aproximar as distribuições dos dois domínios. Esse modelo pode ser generativo (como as GANs, que permitem gerar amostras que seguem a distribuição estatística do domínio alvo a partir de uma entrada do domínio de origem ou um padrão de ruído) ou não generativo (que buscam treinar um extrator de características para mapear ambos os domínios para o mesmo espaço).

- Baseadas em reconstrução: buscam reconstruir um dos domínios a partir de entradas do outro domínio, por meio de *autoencoders* ou GANs cíclicas (cujo aprendizado se baseia em um mapeamento bijetivo entre os dois domínios).

Existem, também, técnicas de DA que não utilizam aprendizado profundo. De acordo com Csurka (2017), alguns exemplos incluem: reponderação de instâncias usando métricas estatísticas, adaptação de parâmetros do classificador para ajustar a margem de decisão no domínio alvo, transformação de *features* para um espaço latente com menor discrepância entre os dois domínios, entre outras.

Há grande variedade nas técnicas que podem ser classificadas como DA, mas de forma geral se pode dizer que a principal desvantagem dessas abordagens é a necessidade de um processo de aprendizado entre os dois domínios, o que adiciona complexidade à geração de dados, além da possibilidade de não obter um resultado satisfatório. Em contrapartida, técnicas de DA requerem apenas imagens dos dois domínios, ou seja, não pressupõem o uso de um modelo paramétrico como domínio de origem. Isso possibilita sua aplicação, por exemplo, na adaptação entre dois conjuntos de dados reais, ou até nos casos em que o domínio de origem tem poucos parâmetros, como as cenas de um *video game*.

2.1.4 Randomização de domínio

Diferentemente da adaptação de domínio, que busca aproximar os domínios estatisticamente, a randomização de domínio (DR — *domain randomization*) tem por objetivo aumentar a variedade e a quantidade de dados do domínio de origem, de forma que o modelo se torne robusto o suficiente para funcionar bem no domínio alvo (NIKOLENKO, 2021). Esta técnica tem grande aplicabilidade na produção de conjuntos de dados sintéticos, devido à capacidade de se manipular computacionalmente os parâmetros de uma simulação.

Para visão computacional, DR é frequentemente aplicada para diversificar imagens geradas a partir de modelos 3D: posição e características dos objetos, câmera, fontes de luz, entre outras (ver subseção 2.1.1). De acordo com Tremblay *et al.* (2018), abandona-se o fotorrealismo ao perturbar aleatoriamente o ambiente, como exemplificado na Figura 4, forçando o aprendizado do modelo a focar nas características essenciais da imagem.

Figura 4 – Imagens de cenários com veículos geradas usando randomização de domínio com manipulação agressiva de parâmetros.

Fonte: Adaptado de Tremblay *et al.* (2018).

A estratégia padrão da DR é randomizar os parâmetros do ambiente de forma uniforme dentro de intervalos predefinidos. Os intervalos são, de certa forma, hiperparâmetros que são escolhidos com base no conhecimento do domínio do problema, a fim de que as imagens geradas sejam diversas, mas coerentes. Escolher intervalos muito estreitos pode ser insuficiente para que a transferência do aprendizado seja bem-sucedida; já intervalos muito amplos podem deixar a tarefa de aprendizado mais difícil que o necessário, pois o algoritmo precisa modelar todas as variações arbitrárias enquanto tenta decifrar as dinâmicas da tarefa (JAMES *et al.*, 2019).

Em vários casos, ajustes nos parâmetros de randomização são feitos heuristicamente por tentativa e erro. Por outro lado, avanços recentes buscam otimizar a DR de forma sistemática. Mehta *et al.* (2020) propõem a randomização de domínio ativa (ADR — *active domain randomization*), que busca encontrar uma configuração de randomização por meio de aprendizado por reforço. Já Prakash *et al.* (2019) apresentam o método de randomização de domínio estruturada (SDR — *structured domain randomization*), que leva em consideração as distribuições de probabilidade do problema em questão para gerar imagens sintéticas aleatórias que preservam a estrutura ou contexto do problema.

Com relação aos pontos negativos da DR, o principal é o pré-requisito de um modelo paramétrico para geração de dados, cuja elaboração pode ser dispendiosa. Além disso, pode-se dizer que há um aumento do custo do treinamento, já que é necessário gerar amostras em abundância. Também pode ser difícil definir limites para os parâmetros disponíveis de forma que as amostras geradas permaneçam coerentes com o problema em questão. Mesmo assim, a capacidade de manipular variáveis e formar um conjunto de dados bastante diverso pode ser justificativa suficiente para utilizar DR, especialmente em aplicações nas quais há dificuldade em obter dados reais.

2.1.5 *Data augmentation*

Shorten e Khoshgoftaar (2019) descrevem *data augmentation* como um conjunto de técnicas que inflam artificialmente um *dataset* de treinamento por meio de transformações, com o objetivo de combater *overfitting*. Essas técnicas partem do pressuposto de que é possível extrair mais informações do conjunto de dados original ao modificá-lo de alguma forma. Algumas técnicas de *data augmentation* em imagens incluem:

- espelhamento (mais comum na direção horizontal), rotação, translação e escala;
- recorte de uma região da imagem;
- alterações de cor: conversão para escala de cinza, ajustes de brilho, contraste, saturação, matiz, entre outros;
- injeção de ruído, normalmente gerado a partir de uma distribuição gaussiana;

- aplicação de filtros, como borramento ou *unsharp masking*.

Data augmentation se distingue da geração de dados sintéticos pelo fato de que a primeira modifica dados já existentes, enquanto que a segunda cria novos dados (NIKOLENKO, 2021). Apesar disso, ambas podem ser aplicadas em conjunto. Wood *et al.* (2021), por exemplo, aplicaram rotações, transformações de perspectiva, borramentos, modulações de brilho e contraste, adição de ruído e conversão para escala de cinza para ampliar o conjunto de dados sintéticos, argumentando que essas manipulações são especialmente importantes em imagens sintéticas por serem livres de imperfeição.

Quando se utiliza *data augmentation*, deve-se considerar transformações que são coerentes com o problema em questão, selecionando também limites apropriados para cada uma. Por exemplo, se o problema envolve imagens de objetos sempre em posição aproximadamente vertical, pode-se evitar fazer transformações de rotação com ângulos acima de um certo valor, pois imagens com essas características não serão alimentadas ao sistema em tempo de inferência.

2.1.6 Modelos pré-treinados em grandes conjuntos de dados

Uma forma de transferência de aprendizado comum em *deep learning* é utilizar camadas de modelos pré-treinados em *datasets* extensos e variados, como ImageNet (DENG *et al.*, 2009), PASCAL VOC (EVERINGHAM *et al.*, 2010) e COCO (LIN *et al.*, 2014). Devido à grande quantidade de dados, as arquiteturas desses modelos frequentemente possuem muitas camadas e parâmetros, necessitando de recursos computacionais elevados para o treinamento. Em grande parte das demais aplicações, seja por falta de dados ou de recursos computacionais, treinar um modelo *from scratch* (inicializado com pesos aleatórios) utilizando essas arquiteturas não é viável; daí surge a ideia de utilizar os pesos pré-treinados.

A intuição dessa técnica é que as *features* extraídas por um modelo pré-treinado em um conjunto de dados suficientemente variado podem generalizar para domínios não vistos durante o treinamento, podendo assim ser aproveitadas para outras tarefas (HAN *et al.*, 2021b). As primeiras camadas de um modelo tendem a identificar padrões mais simples e/ou grosseiros, enquanto que as características de alto nível (que têm alguma semântica) são identificadas por camadas mais profundas (YOSINSKI *et al.*, 2014). As últimas camadas do modelo são especializadas para sua tarefa final, utilizando as *features* extraídas para gerar uma saída (por exemplo, uma classificação).

Para usar um modelo pré-treinado em outra aplicação, em geral removem-se as camadas finais, substituindo-as pela arquitetura da aplicação desejada. Em outras palavras, utiliza-se o extrator de *features* do modelo pré-treinado como entrada da arquitetura da aplicação. O treinamento pode ser feito de diversas formas, entre as quais estão o congelamento de pesos das camadas do modelo pré-treinado, treinamento do modelo como um todo com taxa de apren-

dizado reduzida (ajuste fino), ou uma combinação de ambas. Hinterstoisser *et al.* (2019), por exemplo, adaptaram modelos de detecção de objetos pré-treinados em imagens reais para outros domínios congelando o extrator de *features* e treinando as demais camadas apenas com imagens sintéticas.

A maior desvantagem dessa abordagem é que os modelos pré-treinados são normalmente disponibilizados em arquiteturas padronizadas, as quais frequentemente são maiores ou mais complexas que o necessário para resolver problemas específicos. O uso de uma rede superdimensionada pode impactar o tempo de inferência no sistema final, além da possibilidade de elevar os requisitos computacionais. Apesar disso, o treinamento de uma rede especializada, mesmo que menor, pode exigir uma quantidade e variedade de dados que simplesmente não está disponível, fazendo com que o modelo pré-treinado seja o fator que habilita a solução para o problema.

2.1.7 Ajuste fino em dados reais

Na subseção 2.1.6, introduziu-se o conceito de ajuste fino (*fine-tuning*) a partir de um modelo treinado sobre dados reais de um domínio de origem extenso (como o *dataset* ImageNet). A técnica de ajuste fino também pode ser aplicada para transferir o aprendizado do domínio de dados sintéticos para o domínio real da aplicação desejada.

Dependendo das características da aplicação, os domínios sintético e real podem ter diferenças que não são trivialmente tratadas com o melhoramento do realismo da simulação ou o uso de DR (EBADI *et al.*, 2021). Uma forma de tratar o *domain gap* é, após finalizar o treinamento do modelo com dados sintéticos, continuar o treinamento usando imagens reais do domínio alvo. Isso requer dados reais rotulados, embora em menor quantidade do que seria necessário para treinar o modelo completamente sobre dados reais. O ajuste fino foi estudado por Nowruzi *et al.* (2019), cujos experimentos alcançaram melhores resultados com essa técnica do que com o treinamento misto (uso de imagens sintéticas e reais simultaneamente).

2.2 Redes neurais convolucionais para classificação de imagens

Além das técnicas utilizadas para a geração de dados e promoção do aprendizado entre domínios distintos, outro ponto importante para a elaboração de uma solução de visão computacional é a definição da abordagem a utilizar para efetivamente resolver o problema, ou seja, extrair as informações relevantes das imagens e gerar a saída desejada. As abordagens tradicionais utilizam técnicas baseadas em descritores, tais como HOG (DALAL; TRIGGS, 2005) e SIFT (LOWE, 2004), e classificadores, como SVM (CORTES; VAPNIK, 1995), nas quais a extração de *features* e a classificação são feitas em etapas separadas. Por conta disso, cabe a um especialista garantir que as características extraídas sejam relevantes para o problema

em questão. Segundo O'Mahony *et al.* (2020), com o melhoramento da capacidade dos computadores e periféricos modernos, as abordagens baseadas em *deep learning* se tornaram o estado da arte para a classificação de imagens, já que as redes neurais são treinadas ao invés de programadas, exigindo menos conhecimento de um especialista no problema, e também por conseguirem aproveitar o grande volume de dados disponível na atualidade.

O'Mahony *et al.* (2020) acrescentam que *deep learning* introduziu o conceito de aprendizado de ponta a ponta, em que simplesmente se apresenta à máquina um conjunto de dados rotulado com as classes de objetos presentes em cada imagem. Assim, no próprio processo de treinamento, a rede neural descobre os padrões subjacentes e automaticamente forma as *features* que melhor descrevem cada classe. Adicionalmente, modelos de redes neurais convolucionais têm maior flexibilidade, pois podem ser retreinados em um conjunto de dados diferente para outro caso de uso.

Nesta seção, são descritas as características de algumas arquiteturas de redes convolucionais renomadas que foram exploradas no presente trabalho. Essas arquiteturas, em geral, foram desenvolvidas como soluções para problemas *benchmark* na área de visão computacional, tais como a classificação de imagens e detecção de objetos no conjunto de dados ImageNet. O subconjunto mais utilizado desse *dataset* é o utilizado no desafio *ImageNet Large-Scale Visual Recognition Challenge* (ILSVRC) (RUSSAKOVSKY *et al.*, 2015), que contém cerca de 1,3 milhões de imagens de treinamento, 50 mil imagens de validação e 100 mil imagens de teste rotuladas por humanos e classificadas em 1.000 categorias. O uso de arquiteturas populares facilita a especialização sobre modelos pré-treinados, já que os pesos são frequentemente disponibilizados em ferramentas de desenvolvimento.

A compreensão desta seção pressupõe o conhecimento sobre termos e conceitos básicos da área de *deep learning*, tais como tensor, camadas, função de ativação, convolução, filtros/*kernels*, *pooling*, *feature maps*, entre outros, além do processo de treinamento de uma rede neural convolucional. Maiores detalhes podem ser consultados em Goodfellow, Bengio e Courville (2016).

2.2.1 Inception

A arquitetura Inception foi desenvolvida por Szegedy *et al.* (2015) em participação na competição ILSVRC 2014, na qual foi vencedora com a submissão denominada GoogLeNet. Uma das contribuições proporcionadas por essa arquitetura é a utilização mais eficiente dos recursos computacionais na rede neural em comparação com as arquiteturas convolucionais predecessoras, conseguindo aumentar a largura e profundidade da rede sem impactar dramaticamente a quantidade de parâmetros. Esta arquitetura é composta por "módulos Inception" encadeados (Figura 5), que realizam, para uma mesma entrada, operações de convolução bidimensional com filtros de tamanhos 1x1, 3x3 e 5x5 e uma operação de *pooling*, cujas saídas são concatenadas em um único vetor. Para evitar um aumento expressivo na quantidade de

parâmetros, também são utilizadas camadas convolucionais 1x1 antes das operações de convolução 3x3 e 5x5 e após a operação de *pooling*, já que a convolução 1x1 possibilita definir explicitamente o número de *feature maps* apresentados à camada seguinte.

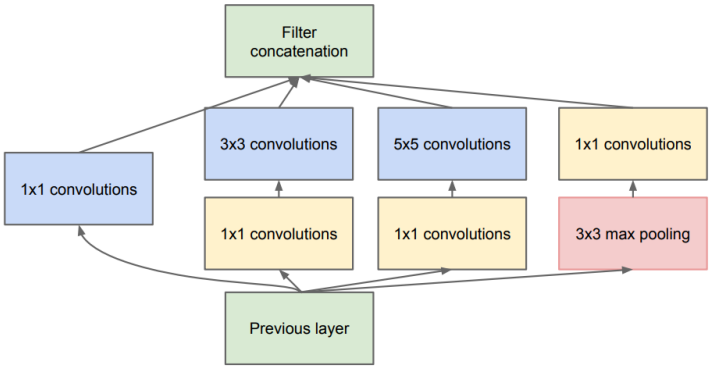


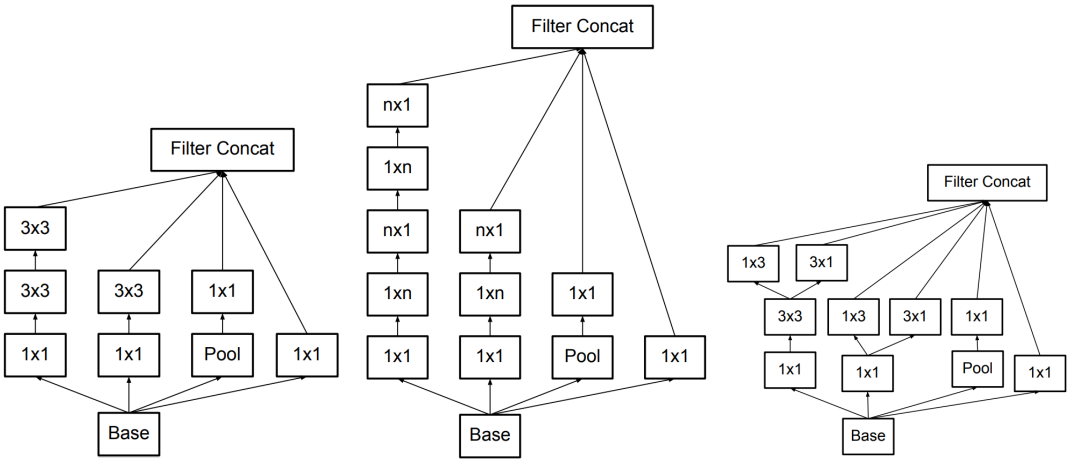
Figura 5 – Módulo Inception com redução de dimensão.
Fonte: Szegedy *et al.* (2015).

Posteriormente, Szegedy *et al.* (2016) realizaram mais otimizações no módulo Inception, tendo observado que o custo computacional das convoluções aumenta desproporcionalmente conforme se aumenta o tamanho dos filtros. A primeira ideia para otimização é que uma convolução com maior tamanho de filtro (por exemplo, 5x5) pode ser fatorada em uma sequência de convoluções menores (por exemplo, duas convoluções 3x3 encadeadas), reduzindo a quantidade total de parâmetros. A segunda ideia é que uma convolução 3x3 pode ser fatorada em duas convoluções assimétricas em sequência: uma 3x1 e outra 1x3, também reduzindo o número de parâmetros. Com isso, os autores propuseram três novos módulos Inception, ilustrados na Figura 6, utilizados para construir uma nova arquitetura, denominada Inception-V2, superando os resultados da anterior. Outros melhoramentos, como o uso de *batch normalization* (IOFFE; SZEGEDY, 2015) e *label-smoothing regularization* (SZEGEDY *et al.*, 2016) melhoraram ainda mais o resultado — essa última iteração foi nomeada Inception-V3.

Neste trabalho, utilizou-se um módulo Inception como parte de uma das arquiteturas descritas na seção 4.9. As arquiteturas Inception completas não foram avaliadas.

2.2.2 DenseNet

A arquitetura DenseNet, proposta por Huang *et al.* (2017), foi desenvolvida como forma de mitigar o problema de dissipação do gradiente frequentemente observado em modelos profundos, no qual a derivada parcial do erro se torna pequena ao ponto de inibir o treinamento. Em termos práticos, isso simboliza que a informação sobre a entrada se perde conforme passa por muitas camadas. Seguindo essa intuição, os autores buscaram melhorar o fluxo de informação pela rede por meio da criação de novas conexões entre camadas não imediatamente subsequentes (*skip connections*).



(a) **(b)** **(c)**

Figura 6 – Módulos Inception com fatoração de convoluções.
Fonte: Szegedy *et al.* (2016).

Nas arquiteturas convolucionais tradicionais, cada camada de convolução é conectada apenas à camada subsequente. Já na arquitetura DenseNet, a saída de camada convolucional é conectada a todas as camadas subsequentes com a mesma dimensão de *feature map*. Em outras palavras, cada camada recebe entrada de todas as camadas anteriores (concatenadas) e fornece sua saída para todas as camadas seguintes. Esse conjunto de camadas interconectadas foi denominado um “bloco denso”, ilustrado na Figura 7.

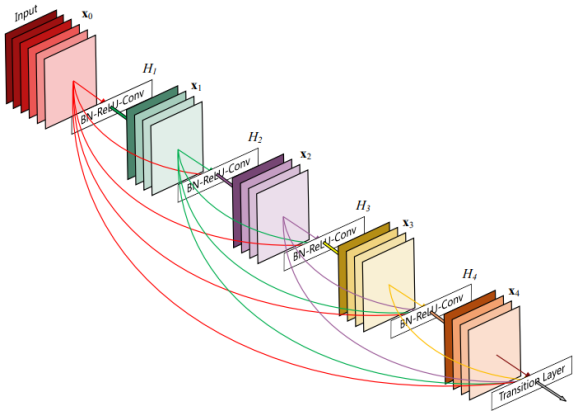


Figura 7 – Um bloco denso com cinco camadas convolucionais.
Fonte: Huang *et al.* (2017).

A arquitetura DenseNet é formada por vários blocos densos ligados por camadas de transição, que reduzem a dimensão da saída de cada bloco por operações de convolução e *pooling*, como exemplificado na Figura 8. No trabalho de Huang *et al.* (2017), são apresentadas quatro instâncias da arquitetura, nomeadas de acordo com o número total de camadas: DenseNet-121, DenseNet-169, DenseNet-201 e DenseNet-264.

Como cada camada recebe como entrada os *feature maps* de todas as camadas anteriores, essa arquitetura promove a reutilização de *features*, melhorando a eficiência no uso de parâmetros. Assim, o número de filtros por camada em uma rede DenseNet tende a ser infe-

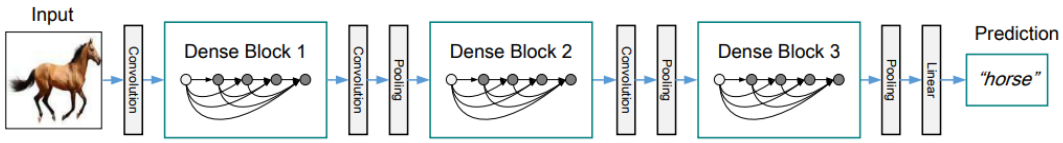


Figura 8 – Uma DenseNet com três blocos densos.
Fonte: Huang *et al.* (2017).

rior às alternativas. Experimentalmente, os autores obtiveram desempenhos comparáveis (e em alguns casos superiores) a arquiteturas residuais com mais do que o dobro da quantidade de parâmetros.

2.2.3 MobileNetV2

A arquitetura MobileNetV2 (SANDLER *et al.*, 2018) foi especialmente desenhada para execução em equipamentos móveis e com recursos limitados, como plataformas embarcadas. Por conta disso, uma das premissas na sua elaboração foi minimizar o número de operações e o uso de memória, mantendo a acurácia elevada.

Para alcançar esses objetivos, a arquitetura utiliza módulos denominados "blocos residuais invertidos" como o ilustrado na Figura 9. A entrada é, primeiro, expandida por uma convolução 1x1, seguida por uma convolução separável em profundidade (*depthwise separable convolution*) 3x3, e depois submetida a uma compressão com uma convolução 1x1. À saída final do bloco, soma-se a entrada da primeira camada, permitindo que a informação não modificada pelo bloco convolucional siga para as próximas camadas da rede. Esse bloco é chamado "residual invertido" porque, no bloco residual tradicional, inicia-se com a compressão e se finaliza com a expansão, e a conexão residual é feita entre os tensores de maior profundidade. Já no bloco residual invertido, a conexão residual é feita entre os *bottlenecks* ("gargalos"), o que é justificado pelos autores pela intuição de que eles têm toda a informação necessária.

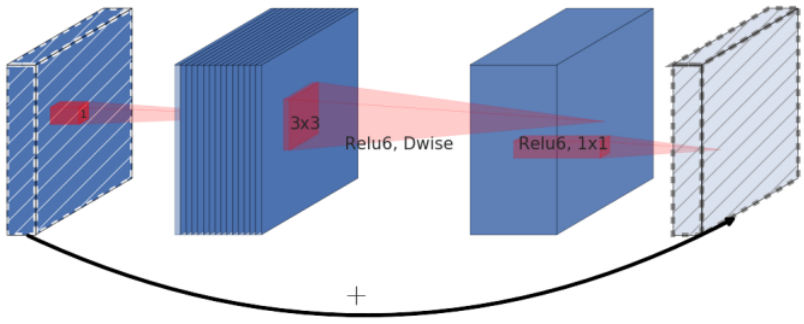


Figura 9 – Bloco residual invertido utilizado na arquitetura MobileNetV2.
Fonte: Sandler *et al.* (2018).

A operação de convolução separável em profundidade é uma forma de fatorar a convolução em duas camadas: a primeira aplica um único filtro convolucional por canal de entrada (o

que é menos custoso que fazer uma convolução completa), e a segunda realiza uma convolução 1x1 para obter combinações lineares dos canais de saída da primeira camada. Sandler *et al.* (2018) argumentam que uma convolução separável em profundidade de dimensão 3x3 é de 8 a 9 vezes menos custosa que uma convolução 3x3 regular, com apenas uma pequena redução de acurácia. Além disso, nas camadas de *bottleneck*, utiliza-se a função de ativação linear, já que não-linearidades (como a função ReLU) causariam a perda de informação. A arquitetura final, denominada MobileNetV2, é composta por uma camada inicial de convolução com 32 filtros seguida por 19 blocos residuais invertidos.

2.2.4 EfficientNet

A família de redes convolucionais EfficientNet tem origem no trabalho de Tan e Le (2019), que investigou a escalabilidade de arquiteturas para obter maior acurácia. Um exemplo já exposto na subseção 2.2.2 é a DenseNet, para a qual Huang *et al.* (2017) apresentaram quatro configurações em sua publicação original, diferindo pelo número de camadas. As três características mais frequentemente modificadas para se aumentar uma arquitetura são a profundidade (número de camadas), a largura (número de filtros por camada) e a resolução (dimensões da imagem). Tan e Le (2019) observaram empiricamente que, para obter os melhores resultados, é crítico manter um equilíbrio entre essas três características, e que esse equilíbrio é atingido ao aumentá-las com uma razão constante. Assim, os autores propuseram um método de escalabilidade composta cujos coeficientes são determinados a partir de uma busca em grade a partir de um modelo pequeno original.

Para demonstrar a solução de escalabilidade, Tan e Le (2019) antes criaram uma arquitetura *baseline* por meio de um método de busca automatizada, otimizando a acurácia e a taxa de operações de ponto flutuante por segundo (FLOPS). A rede resultante, à qual se deu o nome EfficientNetB0, é composta por vários blocos similares ao bloco *bottleneck* da rede MobileNetV2 (subseção 2.2.3), com variações no número e dimensões dos filtros, número de camadas e outras operações adicionais.

Utilizando uma busca em grade, os autores encontraram os três coeficientes de escalabilidade da rede base, que determinam a que proporção se deve aumentar a profundidade, a largura e a resolução da rede. Com isso, foram produzidas outras sete arquiteturas (B1 a B7, da menor para a maior). Experimentos de classificação no ImageNet demonstraram que as redes EfficientNet, em geral, necessitam de uma ordem de grandeza a menos em questão de parâmetros e FLOPS para alcançar acurácia equivalente a outras arquiteturas convolucionais.

3 TRABALHOS RELACIONADOS

Neste capítulo, apresentam-se trabalhos relacionados a esta pesquisa, dividindo-os em dois grandes grupos: na seção 3.1, trabalhos no tema de reconhecimento do estado de chaves seccionadoras, e na seção 3.2, trabalhos que fazem uso de dados sintéticos em aplicações diversas.

3.1 Reconhecimento do estado de chaves seccionadoras

Pesquisas recentes mostram avanços no uso do aprendizado profundo em subestações e outras áreas do setor elétrico, incluindo aplicações tais como predição ou diagnóstico de falhas e defeitos (WANG *et al.*, 2022b; HAN *et al.*, 2021a; DAVARI; AKBARIZADEH; MASHHOUR, 2022; MOGOS; LIANG; CHUNG, 2023; CHEN; WAN; DOU, 2022; HUANG *et al.*, 2022; ZHOU *et al.*, 2023; WANG *et al.*, 2022a) e detecção de equipamentos (ZHENG *et al.*, 2022; OU *et al.*, 2023; ZHAO *et al.*, 2023), utilizando tanto imagens do espectro visível quanto do infravermelho e ultravioleta. Todavia, um problema recorrente reportado em muitos trabalhos é o desbalanceamento de dados, resultante do fato de que os equipamentos de uma subestação permanecem em um estado normal pela maior parte do tempo. Zheng *et al.* (2022), por exemplo, relatam não conseguir implementar uma solução de detecção de falhas devido à falta de imagens de equipamentos em estado de falha. Outros estudos lidam com essa dificuldade de diversas formas, tais como adaptação de domínio (WANG *et al.*, 2022b), *transfer learning* (WANG *et al.*, 2022b; DAVARI; AKBARIZADEH; MASHHOUR, 2022), sobreamostragem e dados sintéticos (MOGOS; LIANG; CHUNG, 2023; CHEN; WAN; DOU, 2022; WANG *et al.*, 2022a), detecção de anomalias (MOGOS; LIANG; CHUNG, 2023) e arquiteturas específicas de redes neurais (ZHOU *et al.*, 2023). No presente trabalho, o desbalanceamento de dados é combatido com imagens sintéticas geradas a partir de modelos 3D de subestações.

Com relação ao reconhecimento do estado de chaves seccionadoras, alguns estudos exploram o uso de sensores instalados fisicamente em cada chave (MOK *et al.*, 2019; CHEN *et al.*, 2022; SEMEDO; OLIVEIRA; CARDOSO, 2014; WERNECK; ALLIL, 2019). Embora essas estratégias sejam eficazes, elas não são facilmente comissionadas em subestações em funcionamento, já que, além dos riscos de segurança, as chaves precisariam ser manipuladas para a instalação dos sensores. A manutenção dos sensores também é uma notável desvantagem dessas soluções, pois, como as manobras devem ser agendadas com antecedência, o mau funcionamento de um sensor poderia deixar o sistema inoperante por longos períodos. Além disso, há desafios adicionais em questões como coleta de energia (*energy harvesting*), blindagem eletromagnética e comunicação de dados.

Como alternativa, outra linha de pesquisa avalia soluções não invasivas baseadas em câmeras. Algumas estratégias envolvem algoritmos clássicos de processamento de imagens, tais como detecção de bordas e retas, seguidos por uma etapa de cálculo de ângulo para de-

terminar o estado da chave (LU *et al.*, 2015; TENG *et al.*, 2019; FU *et al.*, 2019). Kai *et al.* (2021) apresentam uma abordagem mais elaborada, combinando imagens do espectro visível e infravermelho e usando um algoritmo evolucionário para segmentação, mas a classificação final ainda é realizada por cálculo de ângulo entre elementos geométricos. Essas soluções necessitam de enquadramentos de imagem que mostrem a chave com clareza, sem estruturas confusas no fundo da imagem, o que dificilmente é possível em um ambiente de subestação. Ademais, elas exploram propriedades geométricas de tipos específicos de chaves. Chen *et al.* (2017) tratam o problema do fundo por meio de descritores HOG e redução de dimensionalidade com LDA (*Linear Discriminant Analysis*), encontrando sub-imagens contendo apenas o contato da chave, e posteriormente classificando-as com uma SVM. Embora a abordagem tenha se mostrado eficaz no cenário particular do trabalho citado, ela explora características de um tipo específico de chave, e ainda pode ser afetada por objetos de fundo com formas retilíneas que apareçam atrás do contato. Essas limitações não permitem uma comparação direta entre essas soluções e a abordagem do presente trabalho, já que elas não se aplicam ao cenário considerado nesta pesquisa, que pode ter elementos de fundo indeterminados, além de buscar uma solução comum para qualquer tipo de chave.

Abordagens baseadas em aprendizado profundo visam lidar com esses problemas com uma ênfase nos dados: ao apresentar uma grande quantidade de amostras de imagem a uma CNN, o modelo aprende a extrair as características que são relevantes para a classificação. Wang (2018) propõe o uso de uma CNN para reconhecer o estado de chaves seccionadoras a partir de fotos. Para minimizar o impacto do pequeno *dataset*, o autor utiliza *transfer learning*, obtendo uma acurácia de 91,3%. Entretanto, o trabalho citado considerou um conjunto de dados com poucos enquadramentos e tipos de chaves. No presente trabalho, também são utilizadas CNNs para classificação, mas os modelos são treinados sobre dados sintéticos devido ao problema do desbalanceamento, a fim de obter modelos que funcionem para múltiplos pontos de vista.

Quan *et al.* (2022) usam uma estratégia baseada em detecção de objetos com uma CNN, treinada para encontrar a parte móvel e os dois contatos de chaves seccionadoras de dupla abertura. O estado da chave é estimado a partir da distância euclidiana mínima (na imagem, em 2D) entre o braço e os contatos da chave. Os autores reportam uma taxa de acerto de 91,12%, mas essa abordagem não é genericamente aplicável a todos os tipos de chave, e pode ser impactada por perspectivas desfavoráveis de câmera.

Le, Pham e Phung (2022) também fazem a detecção de objetos com CNN para encontrar chaves na imagem. Foram definidas classes separadas para chaves abertas e chaves fechadas, embutindo a lógica do reconhecimento de estado no próprio modelo de detecção. A abordagem é baseada em uma visão de baixo para cima, a fim de deixar o fundo da imagem o mais limpo possível. Um conjunto de apenas 432 imagens foi coletado manualmente e dividido em partições de treinamento, validação e teste, sendo que as imagens de treinamento foram submetidas a *data augmentation*. O modelo treinado foi capaz de localizar e reconhecer

chaves abertas e fechadas nas imagens com uma *Mean Average Precision* (mAP) de 0,9974; já quanto à classificação, a acurácia do sistema não é relatada explicitamente, mas a elevada mAP e o pequeno número de exemplos de teste permitem inferir que chegou próxima a 100%. Entretanto, os autores relatam que o modelo produziu valores baixos de confiança em imagens que continham alguns objetos de fundo com formas similares à chave. Além disso, quase todos os exemplos mostrados no trabalho citado são de dias nublados — em um dia ensolarado, o ponto de vista escolhido faria com que as chaves aparecessem muito escuras nas imagens, podendo até ofuscá-la completamente. No presente trabalho, a abordagem inclui maior variedade de pontos de vista, além da inclusão de fundos aleatórios nas imagens sintéticas, para guiar o modelo a ignorar o fundo da imagem.

De forma geral, há poucas iniciativas relacionadas ao presente trabalho para realizar o reconhecimento do estado de chaves seccionadoras por imagens, e as iniciativas existentes frequentemente utilizam simplificações do domínio do problema para minimizar a exposição a exemplos desafiadores, como fundos confusos, pontos de vista variados, tipos distintos de chaves, entre outros. Essas situações existem na prática e precisam ser tratadas por um sistema cujo objetivo é fazer parte da automação de subestações.

3.2 Uso de dados sintéticos no treinamento de modelos

Técnicas de ampliação de conjuntos de dados (*data augmentation*) são parte crucial de muitos trabalhos de visão computacional da atualidade. Por meio de transformações como rotações, recortes, modificações de cor, entre outras, é possível produzir novos dados a partir de dados já existentes, contribuindo para a diversidade do *dataset*. Mais recentemente, foram desenvolvidos métodos mais sofisticados que permitem produzir novos dados a partir de outras fontes, como simulações, modelos 3D, modelos generativos, entre outras. Nesta seção, são apresentados alguns trabalhos que pertencem a essa categoria, ou seja, utilizam dados sintéticos em tarefas de inteligência artificial.

3.2.1 Métricas utilizadas nos trabalhos de referência

Antes de discutir os trabalhos de referência, é importante definir algumas métricas utilizadas para a avaliação dos modelos. Vários trabalhos discutidos neste capítulo tratam de problemas de detecção ou localização de objetos em imagens. A saída de um modelo de detecção de objetos para uma imagem de entrada é um conjunto de detecções, cada uma usualmente composta por uma classificação (a categoria do objeto detectado), uma *bounding box* (retângulo que define a posição e o tamanho do objeto na imagem) e um valor de confiança (normalmente entre 0 e 1). Em uma etapa de filtragem das saídas, define-se um limiar de confiança para as

detecções aceitas — limiares altos possivelmente farão com que o número de detecções seja menor, mas elas terão maior probabilidade de serem corretas.

A avaliação de modelos de detecção de objetos frequentemente utiliza a métrica de interseção sobre união (IoU — *Intersection over Union*), definida como a razão entre a área de interseção e a área de união entre a *bounding box* encontrada na detecção e a *bounding box* verdadeira rotulada (*ground truth*). Durante a avaliação, define-se um limiar para a IoU (que não é o limiar de confiança citado anteriormente): se a IoU calculada for igual ou superior ao limiar, considera-se que a detecção é um verdadeiro positivo; caso contrário, trata-se de um falso positivo. Também são considerados falsos positivos os casos em que o modelo detecta um objeto em uma região incorreta, e falsos negativos são contabilizados quando o modelo deixou de detectar algum objeto que havia sido rotulado. É comum a utilização de limiares de IoU como 0,5 (valor adotado pelo desafio PASCAL VOC (EVERINGHAM *et al.*, 2010)) ou 0,75 (valor adotado como "métrica estrita" para o *dataset* COCO (LIN *et al.*, 2014)).

Com a contagem de verdadeiros positivos, falsos positivos e falsos negativos, pode-se calcular os valores de precisão e *recall*, que são formas de avaliar o desempenho de um modelo estatístico. Sendo TP o número de verdadeiros positivos, FP o número de falsos positivos e FN o número de falsos negativos, a precisão é calculada por $\frac{TP}{TP+FP}$, representando a fração das instâncias recuperadas que é relevante. Já o *recall* é calculado por $\frac{TP}{TP+FN}$, representando a fração das instâncias relevantes que foi recuperada (GOODFELLOW; BENGIO; COURVILLE, 2016). Um modelo perfeito tem precisão e *recall* iguais a 100%, mas, na prática, é comum observar uma relação inversamente proporcional entre eles. Por conta disso, outra estatística, o F_1-score (também conhecido como F-measure), une as duas métricas de forma equilibrada, sendo definido como a média harmônica entre a precisão (p) e o *recall* (r), ou seja, $\frac{2pr}{p+r}$.

Como o número de detecções é afetado pelo limiar de confiança definido anteriormente, os valores de precisão e *recall* são funções desse limiar: ao aumentá-lo, a precisão normalmente sobe e o *recall* cai; ao reduzí-lo, ocorre o oposto. Para avaliar o desempenho do modelo de forma geral, fixa-se um limiar de IoU e se altera o limiar de confiança ao longo de todo o seu domínio, construindo, assim, uma curva de precisão *versus recall*. A métrica de precisão média (AP — *average precision*), que sintetiza o *tradeoff* entre precisão e *recall* para uma determinada classe, é calculada com uma aproximação numérica da área abaixo da curva. Usualmente, essa métrica é denotada como AP@0.5, onde o valor após o símbolo "@" indica o limiar fixo de IoU utilizado.

Entretanto, a definição de um limiar de IoU fixo depende das necessidades do problema — embora para muitos casos gerais uma sobreposição de 50% já seja suficiente para indicar uma detecção bem-sucedida, em outros pode ser importante que haja um alinhamento consideravelmente mais exato das *bounding boxes*, com valores acima de 95%. Para também generalizar a avaliação do modelo com relação ao limiar de IoU, é comum calcular a AP para diversos valores de IoU e fazer a média aritmética entre elas. Na competição COCO (LIN *et al.*, 2014), por exemplo, utiliza-se a média das APs para valores de IoU entre 0,5 e 0,95, com

passos de 0,05. Alguns autores denotam essa métrica como AP@[.5:.05:.95], ou simplesmente AP@[.5:.95].

Em modelos que detectam objetos de múltiplas classes, as APs das diversas classes podem, ainda, ser combinadas em uma única métrica por meio de simples média aritmética. Essa métrica é denominada *mean average precision* (mAP), e também pode ser denotada com o símbolo "@" para declarar o limiar de IoU utilizado.

3.2.2 Aplicações de dados sintéticos na literatura

O pacote de software "Unity Perception" (BORKMAN *et al.*, 2021) é um conjunto de ferramentas para geração de conjuntos de dados sintéticos cujas funcionalidades incluem renderização de objetos 3D, randomização de parâmetros e anotações perfeitas. Para demonstrar as técnicas, os autores aplicaram a ferramenta para o problema de detecção de produtos de mercado em meio a outros produtos, utilizando a arquitetura Faster R-CNN com *backbone* ResNet-50. Foram preparados dois conjuntos de dados, um contendo 1.267 imagens reais (particionadas em 60% para treinamento, 20% para validação e 20% para teste) e outro contendo 400.000 imagens sintéticas (particionadas em 90% para treinamento e 10% para validação), ilustrados na Figura 10. Como linha de base, o modelo foi inicialmente treinado apenas com imagens reais, obtendo mAP@[.5:.95] igual a 0,450. Usando apenas dados sintéticos, a mAP@[.5:.95] foi inferior: 0,381. Adicionando-se uma etapa de ajuste fino com apenas 76 amostras reais, a mAP@[.5:.95] subiu para 0,528. O melhor resultado foi obtido com ajuste fino usando todas as 760 imagens reais da partição de treinamento, obtendo mAP@[.5:.95] igual a 0,684.

(a) Imagem sintética **(b) Imagem real**

Figura 10 – Comparação entre uma imagem gerada a partir de modelos 3D usando Unity Perception e uma imagem real, ambas contendo produtos de mercado.
Fonte: Adaptado de Borkman *et al.* (2021).

Outra aplicação em detecção de objetos é descrita por Hinterstoisser *et al.* (2019). O método de geração de dados sintéticos envolveu a renderização de um modelo 3D de objeto

usando OpenGL[1], variando-se a rotação do objeto e cor da iluminação, com posterior colagem sobre um plano de fundo real contendo outros objetos dispostos de forma desordenada (Figura 11). Para melhor mesclar o objeto virtual com a imagem real, aplicou-se ruído e borramento após a renderização. O processo de geração de imagens também produz as anotações de *bounding box* do objeto automaticamente. Foram testadas as arquiteturas de detecção Faster R-CNN, R-FCN e Mask R-CNN e os extratores de *features* InceptionResNet e ResNet-101, com pesos pré-treinados no conjunto ImageNet. Os modelos foram avaliados em cerca de 20 mil amostras sintéticas de treinamento e 1 mil amostras reais de teste. Os melhores resultados foram obtidos com o modelo Faster R-CNN+InceptionResNet congelando-se os pesos das camadas do extrator de *features*, o qual obteve mAP@[.5:.95] igual a 0,725 usando apenas dados sintéticos. O mesmo modelo, treinado em dados reais, obteve mAP@[.5:.95] igual a 0,764. Os autores também demonstram que treinar o modelo como um todo (sem congelar o extrator de *features*) prejudica o desempenho.

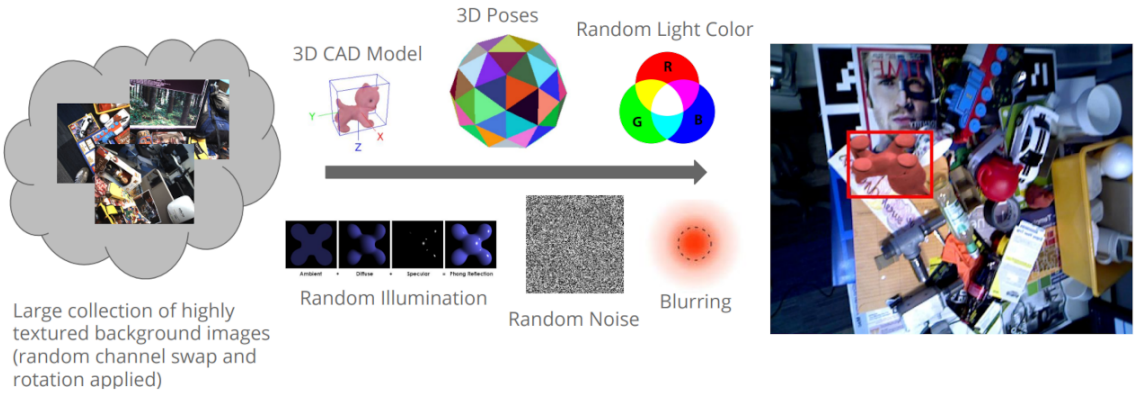


Figura 11 – Processo de geração de imagens sintéticas por meio da renderização de um objeto sobre um plano de fundo real.
Fonte: Hinterstoisser *et al.* (2019).

Ebadi *et al.* (2021) aplicaram dados sintéticos para detecção de pontos de interesse em pessoas usando visão computacional. O gerador de dados desenvolvido pelos autores, chamado “PeopleSansPeople”, cria uma cena virtual com modelos 3D de seres humanos em poses variadas e objetos distratores para produzir imagens sintéticas e anotações, conforme exemplificado na Figura 12. A arquitetura selecionada foi a Keypoint R-CNN com *backbones* ResNet-50 e FPN. Gerou-se um conjunto de 490.000 imagens sintéticas para treinamento e 10.000 para validação, e para fins de comparação, utilizou-se a versão de 2017 do *dataset* COCO. O modelo treinado apenas em imagens reais (32.057 amostras) obteve mAP@[.5:.95] igual a 0,5580, enquanto que o modelo treinado apenas nos dados sintéticos obteve 0,0370, muito aquém do esperado. Após realizar o ajuste fino do modelo com as 32 mil imagens reais, a mAP@[.5:.95] melhorou para 0,6037, superando o *benchmark*. A respeito do baixo desempenho sem o ajuste fino, os autores comentaram que um modelo treinado somente em dados sintéticos gerados com randomização de domínio “ingênua” tem dificuldade em generalizar para o domínio real.

[1] https://www.opengl.org/

Figura 12 – Exemplos de imagens sintéticas de pessoas geradas pelo PeopleSansPeople, contendo anotações automáticas de pontos de interesse.
Fonte: Ebadi *et al.* (2021).

Em uma aplicação semelhante (detecção de pessoas), Tripathi *et al.* (2019) buscaram gerar dados sintéticos por meio de composição 2D a partir de recortes de imagens reais, sem utilizar modelos 3D. Para gerar amostras mais significativas para o treinamento do modelo, os autores utilizaram aprendizado adversário entre três redes convolucionais: um sintetizador, que gera uma transformação afim de 6 parâmetros a partir de duas imagens de entrada (o plano de fundo e o recorte do objeto); um discriminador, que tenta reconhecer se a imagem é real ou sintética; e uma rede alvo, que faz a tarefa de detecção. O sintetizador é otimizado de forma a "enganar" tanto o discriminador quanto a rede alvo, produzindo, assim, composições mais realistas (exemplificadas anteriormente na Figura 2). Para a tarefa final, selecionou-se a arquitetura SSD300 como rede alvo e treinou-se o sistema triplo usando as imagens do PASCAL VOC, obtendo AP@0.5 igual a 0,7961. Em comparação, usando apenas as imagens originais (sem o sintetizador e o discriminador), obteve-se AP@0.5 igual a 0,7893. O ganho de desempenho foi ainda mais significativo para limiares maiores de IoU.

Para a aplicação de detecção de veículos, Gaidon *et al.* (2016) usaram a *engine* de jogos Unity para criar o conjunto de dados sintético "Virtual KITTI", composto por 35 vídeos fotorrealistas com aproximadamente 17 mil quadros de alta resolução, todos com anotações automáticas. Os vídeos são recriações virtuais de cinco vídeos reais do *dataset* "KITTI" (GEIGER *et al.*, 2013) (Figura 13), sendo que os demais foram reservados para teste dos modelos. Foram utilizados dois detectores, um baseado na arquitetura Fast R-CNN e outro baseado em

Processos de Decisão de Markov (MDP) com aprendizado por reforço. Em um primeiro cenário, os autores treinaram os modelos apenas com dados reais, obtendo, entre outras métricas, acurácia de rastreamento multi-objeto (MOTA) igual a 71,9% e 78,1%, respectivamente. No segundo cenário, os modelos foram treinados apenas com dados sintéticos, resultando em MOTAs iguais a 64,3% e 63,7%, inferiores ao cenário anterior. No terceiro cenário, fez-se ajuste fino dos modelos usando os dados reais, obtendo MOTAs iguais a 76,7% e 78,7%, confirmando que o pré-treinamento com dados virtuais melhora o desempenho do modelo. Além disso, os autores identificaram que é necessário utilizar um critério de *early stopping* na etapa de ajuste fino para evitar *overfitting* no conjunto de dados real.

Figura 13 – Quadros do *dataset* real KITTI (à esquerda) e seus equivalentes no Virtual KITTI (à direita).

Fonte: Adaptado de Gaidon *et al.* (2016).

Ainda nesta aplicação, Tremblay *et al.* (2018) conseguiram bons resultados usando apenas dados sintéticos, mas aplicando a técnica de randomização de domínio (DR). O conjunto de dados composto por 100 mil amostras foi gerado utilizando Unreal Engine, variando-se aspectos da cena virtual tais como quantidade e tipo de veículos, objetos distratores, cor e textura dos objetos, posição e angulação da câmera, localização de pontos de luz, imagem de fundo e visibilidade do chão. Alguns exemplos estão ilustrados no Capítulo 2, na Figura 4. Um ponto a se considerar é que os autores intencionalmente não buscaram gerar imagens fiéis à realidade, buscando ao invés disso tratar a diferença entre o domínio real e simulado com uma grande quantidade de variações aleatórias. Foram avaliadas três arquiteturas: Faster R-CNN com Inception ResNet V2, R-FCN e SSD, as quais foram treinadas separadamente no *dataset* Virtual KITTI (VKITTI) e nos dados produzidos pelos autores. Para teste, usaram-se 500 imagens reais escolhidas aleatoriamente do conjunto KITTI. Observou-se que, para as arquiteturas R-FCN e SSD, os dados gerados com DR produziram modelos com melhor desempenho que o VKITTI, enquanto que para a Faster R-CNN observou-se o contrário, embora com pouca diferença (AP@0.5 igual a 0,781 para DR *versus* 0,797 para VKITTI). Além disso, em uma segunda avaliação, os autores inicializaram o modelo Faster R-CNN com pesos treinados no *dataset* COCO e novamente treinaram sobre os dois conjuntos sintéticos. Com o VKITTI, obteve-se AP@0.5 igual a 0,797 (sem melhoria), enquanto que com os dados gerados pelos autores,

obteve-se 0,837, sugerindo que a randomização de domínio tem melhor potencial para *transfer learning*.

Em contraste com a abordagem de DR, Prakash *et al.* (2019) introduzem uma variante denominada randomização de domínio estruturada (SDR), na qual é considerada a probabilidade de ocorrência de eventos de acordo com o contexto. Em outras palavras, os autores aplicaram uma forma de randomização hierárquica de atributos da simulação, sendo que atributos que estão acima na hierarquia determinam o modo com que são randomizados os demais atributos. Na aplicação de detecção de veículos, primeiramente é escolhido aleatoriamente um cenário (urbano, rural etc.), que determina as probabilidades de se posicionar um carro na pista, pessoas na calçada, entre outras propriedades. Para a tarefa de detecção, utilizou-se Faster R-CNN com ResNet V1 pré-treinado no *dataset* ImageNet. O modelo foi treinado em 25 mil imagens geradas com SDR (Figura 14) e testado no *dataset* real KITTI, obtendo AP@0.7 igual a 0,525, em comparação com 0,240 usando apenas DR. Com ajuste fino em apenas 3,3% do conjunto de dados reais (200 amostras), o desempenho subiu para 0,695.

Figura 14 – Imagens de cenários de vias urbanas e rurais geradas com SDR.
Fonte: Adaptado de Prakash *et al.* (2019).

Ainda focando em aplicações veiculares, Sankaranarayanan *et al.* (2018) exploram o problema de segmentação semântica com dados sintéticos. Notando a existência do *domain gap*, elaborou-se uma arquitetura baseada em adaptação de domínio com aprendizado adversário (Figura 15): uma rede F mapeia as imagens de ambos os domínios para *embeddings*; uma rede C gera a saída de segmentação a partir das *embeddings*; uma rede geradora G busca reconstruir a imagem a partir das *embeddings*; e uma rede discriminadora D classifica a imagem reconstruída em real/falsa e origem/alvo. O objetivo da arquitetura é fazer com que a rede F aprenda um mapeamento invariante a domínio para a aplicação desejada. Foram realizados dois experimentos, usando como domínio de origem os *datasets* sintéticos SYNTHIA (ROS *et al.*, 2016) (9.400 imagens) e GTA5 (RICHTER *et al.*, 2016) (24.966 imagens), respectivamente. Em ambos, o domínio alvo foi o *dataset* Cityscapes (CORDTS *et al.*, 2016): usou-se a partição *train* (2.975 imagens) para a adaptação de domínio não supervisionada e a partição *val* (500 imagens) para teste. Para linha de base, treinou-se o modelo apenas no Cityscapes, sem adaptação. No primeiro teste, a interseção sobre união média (mIoU) foi 0,361, *versus* 0,268; já no segundo teste, a mIoU foi 0,371, *versus* 0,296, ou seja, foi constatado um ganho considerável após a adaptação de domínio não supervisionada.

Também para segmentação semântica, Chen *et al.* (2019) aplicam adaptação de domínio de outra forma: uma rede de transformação é usada para transformar imagens sintéticas de entrada em equivalentes com estilo realista (como as apresentadas anteriormente na Figura 3),

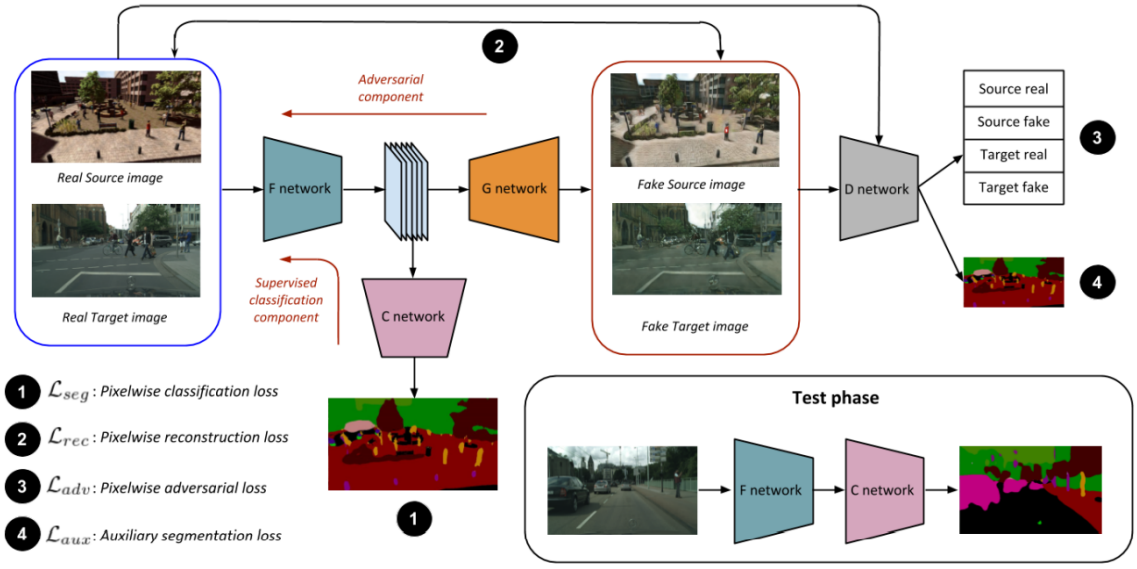


Figura 15 – Arquitetura de um modelo baseado em adaptação de domínio no qual se utiliza um processo de aprendizado adversário para mapear as imagens de entrada (tanto reais quanto sintéticas) para um espaço vetorial comum.
Fonte: Sankaranarayanan *et al.* (2018).

por meio de aprendizado adversário. Para melhorar o desempenho da segmentação e reduzir o *domain gap*, os autores incluíram no treinamento a informação de profundidade (*depth*) dos dados sintéticos, argumentando que a semântica e a geometria estão correlacionadas. Assim, o modelo foi concebido para produzir duas saídas: a segmentação semântica e o mapa de profundidade. Também usou-se um discriminador na saída do modelo para alinhar a rede para ambos os domínios e reter a correlação entre as duas saídas. O primeiro teste foi feito sobre os dados do VKITTI (sintético) e KITTI (real), obtendo mIoU igual a 0,535, *versus* 0,375 sem adaptação. Em seguida o modelo foi avaliado nos *datasets* SYNTHIA (sintético) e Cityscapes (real), atingindo mIoU igual a 0,373, *versus* 0,233, superando também o resultado de Sankaranarayanan *et al.* (2018).

Outros trabalhos conseguiram sucesso sem utilizar imagens reais durante o treinamento. Tobin *et al.* (2017), por exemplo, exploraram a randomização de domínio para gerar imagens sintéticas e treinar um modelo para detecção de objetos em uma aplicação de controle robótico (Figura 16). Uma das justificativas apresentadas para o uso de DR foi a discrepância entre simulações físicas e o mundo real, pois o ajuste de parâmetros para aproximar os dois domínios consome muito tempo e está sujeito a erros. Para a aplicação proposta, os autores utilizaram uma versão modificada da arquitetura VGG-16. A acurácia do detector foi medida experimentalmente na própria aplicação, sendo que o erro médio de detecção (entre vários formatos de objeto) foi de 1,5 cm. Por meio de estudos de ablação, os autores demonstraram que o modelo atingiu desempenho suficiente a partir de 5 mil amostras de treinamento e mostrou melhora até 50 mil amostras. Também foi avaliada a inicialização com pesos pré-treinados no conjunto ImageNet, a qual mostrou-se positiva até 10 mil amostras sintéticas de treinamento, mas além dessa quantidade, não se observou melhora significativa em relação à inicialização aleatória.

Outra questão abordada foi a sensibilidade do modelo em relação às propriedades randomizadas no gerador de dados: observou-se grande degradação de desempenho ao reduzir a quantidade de texturas únicas de objeto, bem como degradações menores ao desabilitar ou restringir outras randomizações.

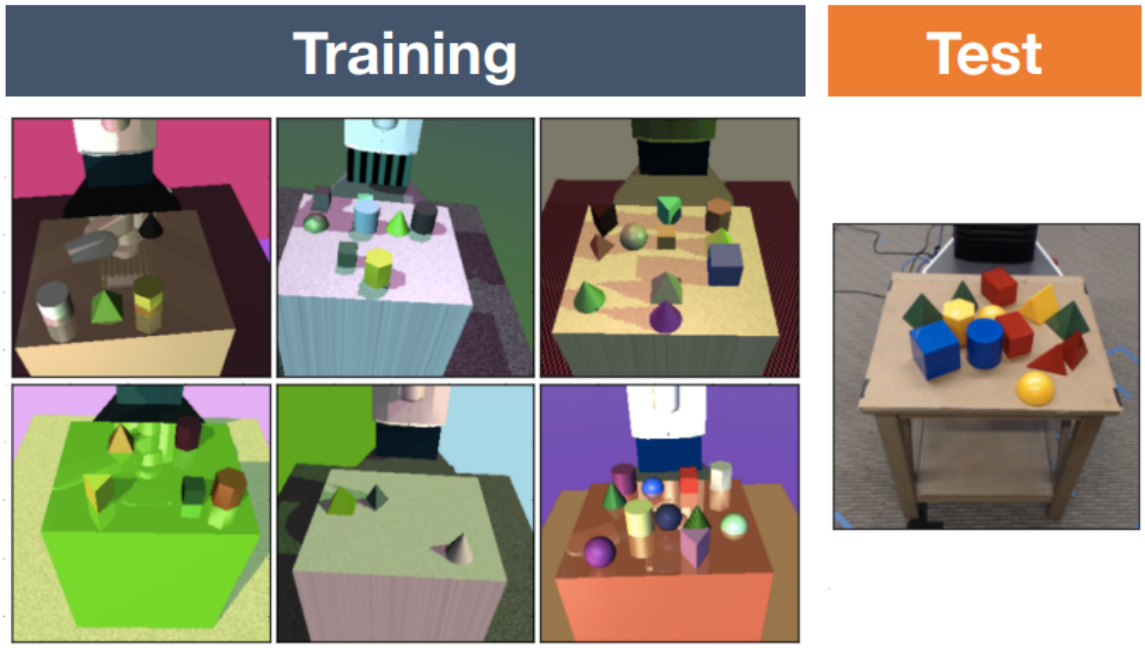


Figura 16 – Exemplos de amostras dos conjuntos de treinamento (sintético) e teste (real) do sistema de detecção de objetos para agarramento robótico.
Fonte: Adaptado de Tobin *et al.* (2017).

O trabalho de James *et al.* (2019) apresenta uma técnica chamada *Randomized-to-Canonical Adaptation Network* (RCAN), que consiste em criar um modelo que traduz imagens reais em uma versão simulada equivalente. O treinamento do modelo é feito apenas com imagens sintéticas geradas randomicamente, aprendendo a adaptá-las a versões canônicas não randomizadas (Figura 17). Os autores utilizaram o método proposto na aplicação de agarramento robótico, em conjunto com o algoritmo QT-Opt, baseado em aprendizado por reforço. O modelo obteve desempenho de 70% com RCAN, em comparação com 37% usando apenas randomização de domínio. Ao fazer ajuste fino com cerca de 5 mil imagens reais, o desempenho subiu para 91%, comparável com um sistema treinado em centenas de milhares de amostras reais.

O trabalho de Wood *et al.* (2021) descreve o uso de dados sintéticos para tarefas de visão computacional em rostos humanos, tais como segmentação semântica de partes do rosto e localização de pontos de interesse. Os sistemas foram treinados em um conjunto de dados contendo 100 mil imagens geradas a partir de um modelo 3D paramétrico de rosto humano. Ao invés de utilizar técnicas de adaptação de domínio para tratar o *domain gap* entre os dados sintéticos e os reais, os autores buscaram essa aproximação por meio de melhorias no realismo e expressividade das imagens geradas sinteticamente, conforme ilustrado na Figura 18. Também aplicou-se *data augmentation* no momento do treinamento, o que, segundo os autores, é espe-

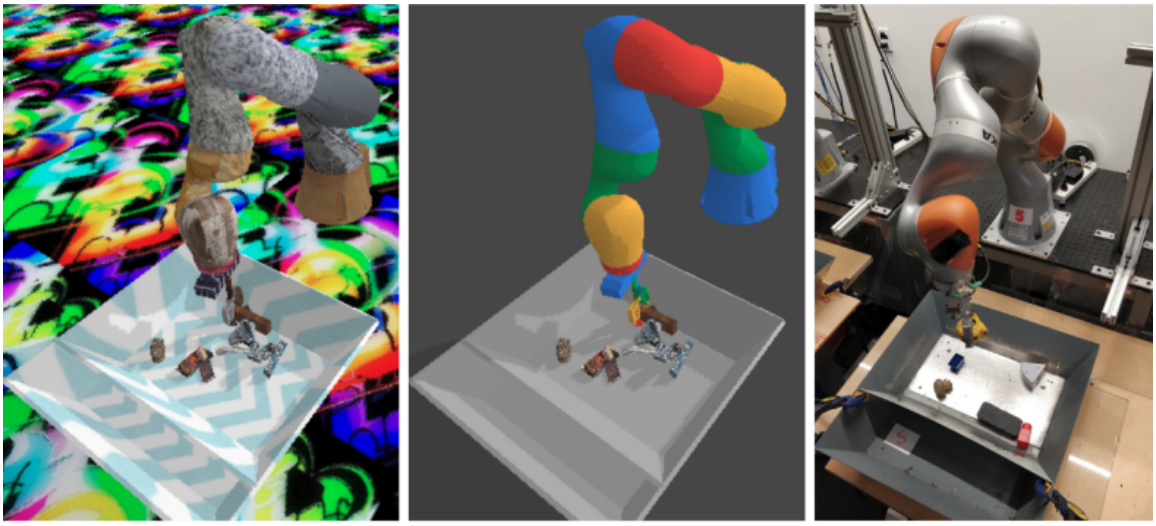


Figura 17 – Exemplos de imagens utilizadas na técnica RCAN, na qual o modelo traduz tanto imagens reais quanto sintéticas (randomizadas) para uma versão canônica equivalente com características simplificadas.

Fonte: James *et al.* (2019).

cialmente importante para imagens sintéticas, que têm poucas imperfeições. Para segmentação semântica, usou-se a arquitetura U-Net com ResNet-18, obtendo F_1-score igual a 0,901 no teste com o conjunto de dados real LaPa (LIU *et al.*, 2020), em comparação com 0,911 do estado da arte. Já para localização de pontos de interesse, usou-se a ResNet-34, que obteve erro médio normalizado (NME — *normalized mean error*) igual a 0,0486 na partição *Challenging* do *dataset* 300-W (SAGONAS *et al.*, 2016), em comparação com 0,0452 do estado da arte. Embora ambos os resultados não tenham superado seus antecessores, é importante ressaltar que os modelos foram treinados apenas sobre dados sintéticos e foram comparados com modelos treinados em grandes quantidades de dados reais.

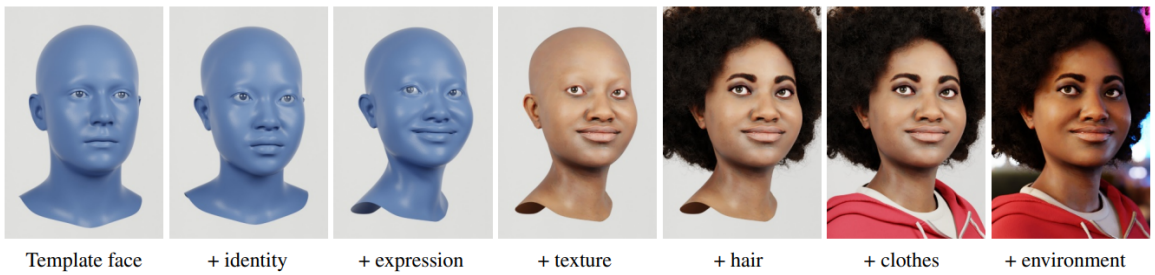


Figura 18 – Geração de imagens sintéticas realistas de rostos humanos com randomização de domínio.

Fonte: Wood *et al.* (2021).

3.2.3 Discussão

Para obter uma melhor visualização das tendências na geração e uso de dados sintéticos em visão computacional, construiu-se a Tabela 1 por meio da identificação de quais técnicas foram aplicadas em cada trabalho enunciado na subseção 3.2.2. Para fins comparativos, o presente trabalho também foi incluído na tabela.

Tabela 1 – Técnicas aplicadas na geração de dados sintéticos em cada trabalho de referência.

	Técnica						
Trabalho	Model. 3D	Comp. 2D	DA	DR	Data augm.	Pesos pré-tr.	Uso de d. reais
Gaidon *et al.* (2016)	●					●	●
Tobin *et al.* (2017)	●			●	①	②	
Sankaranarayanan *et al.* (2018)	●		●		●	●	
Tremblay *et al.* (2018)	●	①		●	●	●	●
Hinterstoisser *et al.* (2019)	●	●		●		●	
Prakash *et al.* (2019)	●			●	●	●	●
Tripathi *et al.* (2019)		●			⊛	●	●
Chen *et al.* (2019)	●		●		①	●	
James *et al.* (2019)	●		⊛	●			●
Borkman *et al.* (2021)	●			●	●	●	●
Wood *et al.* (2021)	●	①		●	●		
Ebadi *et al.* (2021)	●	①		●			●
Este trabalho	●	①		●	●	●	●

● = técnica usada extensivamente
⊛ = a abordagem como um todo pode ser categorizada nesta técnica
① = técnica usada de forma limitada
② = técnica inicialmente usada, mas testes a mostraram desnecessária

Fonte: Autoria própria (2024).

A partir do estudo dos trabalhos correlacionados, conclui-se que a modelagem 3D é a principal técnica de geração de dados sintéticos para problemas de visão computacional. Poucos trabalhos focam em fotorrealismo, possivelmente devido ao tempo e custo adicional — os demais trabalhos priorizaram a randomização de parâmetros. A modelagem 3D é associada em alguns casos à composição 2D para adição de planos de fundo às renderizações de uma cena. Outro uso de composição 2D é a geração de imagens sintéticas por meio de recorte-e-cola de pedaços de imagens reais em outras (TRIPATHI *et al.*, 2019), embora essa técnica seja menos frequente que a renderização.

O principal risco no uso de dados sintéticos é a falha na transferência de aprendizado devido ao *domain gap*, por conta de vieses intrínsecos do processo de geração. Para combater isso, os trabalhos analisados utilizam uma combinação de técnicas, incluindo adaptação

de domínio, randomização de domínio, *data augmentation*, modelos pré-treinados em grandes conjuntos de dados e ajuste fino em dados reais.

A randomização de domínio é a principal vantagem dos dados sintéticos gerados por simulação, porque torna possível gerar uma grande variedade de situações a partir de um único modelo 3D. Trabalhos que usam DR em algumas situações não usam técnicas de *data augmentation*, mas Wood *et al.* (2021) explicitam sua importância para dados sintéticos.

Técnicas de adaptação de domínio dependem de dados do domínio alvo (reais) para aproximar os domínios. Nos trabalhos analisados, DA aparenta não ser frequentemente utilizada em conjunto com DR, embora as duas não sejam mutuamente exclusivas. Uma possível explicação é que, nos trabalhos que usam DR, a função da DA é, de certa forma, coberta pelo ajuste fino em dados reais. Mesmo assim, DA permanece como uma opção nos casos em que a geração de dados sintéticos tem poucos parâmetros variáveis (como em *video games*), ou em aplicações quaisquer que tenham disponibilidade de dados reais, inclusive em conjunto com DR.

A utilização de modelos pré-treinados em *datasets* como ImageNet e COCO é bastante difundida entre os trabalhos. A intuição por trás disso é que o treinamento em dados sintéticos deve focar em aprender características de alto nível. Em geral, o uso dessa técnica parece ser benéfica à transferência de aprendizado, mas na aplicação específica de Tobin *et al.* (2017), a partir de uma certa quantidade de dados sintéticos, os pesos pré-treinados passaram a não trazer vantagem significativa no resultado.

Embora a maior parte dos trabalhos tenha conseguido aplicar os modelos treinados apenas em dados sintéticos no domínio real diretamente, o desempenho nem sempre conseguiu superar o treinamento em dados reais. Diante disso, uma técnica usada frequentemente é o uso de pequenas quantidades de dados reais no treinamento, seja em conjunto com os dados sintéticos ou em uma etapa posterior de ajuste fino. Nota-se que o pré-treinamento em dados sintéticos com posterior ajuste fino em um conjunto de dados real geralmente consegue obter melhores resultados que o treinamento direto sobre o *dataset* real.

Observando-se os trabalhos selecionados de forma geral, é possível perceber uma tendência crescente na utilização de dados sintéticos para tarefas de aprendizado de máquina. Em muitos casos, a demanda por dados sintéticos surge por dificuldades na coleta de dados reais com quantidade ou diversidade suficiente. Isso pode ocorrer devido às próprias características do domínio em questão ou devido a fatores externos, como leis e regulações. Outra razão é o trabalho de rotulagem/anotação manual dos dados, que se torna impeditivamente caro, além de estar sujeito a erros. A vantagem de se utilizar dados sintéticos está na capacidade de gerar grandes quantidades de dados por computador, modificar atributos para obter diversidade de dados e gerar anotações automáticas. Em suma, o uso de dados sintéticos em inteligência artificial mostra-se viável, amparado pelos trabalhos analisados. A capacidade de gerar uma quantidade arbitrária de dados abre oportunidade para aplicações que possivelmente não seriam possíveis apenas com dados reais. Entretanto, o sucesso do uso de dados sintéticos está

condicionado à adoção de alguma técnica para promover a transferência de aprendizado entre os dois domínios.

4 MATERIAIS E MÉTODOS

Para resolver o problema de reconhecimento do estado de chaves seccionadoras, propõe-se uma abordagem baseada em redes neurais convolucionais com a finalidade de classificar amostras de entrada em duas categorias: chave aberta e chave fechada. Com vista ao desbalanceamento de dados resultante da característica de que as chaves permanecem por longos períodos no mesmo estado, a alternativa proposta é a geração de dados sintéticos a partir de um modelo 3D paramétrico da subestação. Dessa forma, as chaves seccionadoras podem ser manipuladas no ambiente virtual, possibilitando a simulação do estado oposto àquele em que normalmente estão. Os modelos devem ser treinados majoritariamente com dados sintéticos, já que a execução de manobras para coleta de dados é uma tarefa de difícil programação, pois exige mobilização de equipes de engenharia para evitar interrupções no fornecimento de energia.

Nas seções a seguir, são detalhados os passos e características da solução proposta.

4.1 Escolha de subestações e chaves seccionadoras

Para o desenvolvimento e teste da solução proposta, foram escolhidas duas subestações de transmissão de níveis de tensão distintos. A primeira foi a Subestação Campo Comprido (SE CCO), localizada em Curitiba, Paraná, no Bairro Orleans, ilustrada na Figura 19. Além de ter uma distribuição bastante heterogênea quanto aos tipos de chaves seccionadoras, a proximidade da subestação à cidade de Curitiba facilita os trabalhos em campo, especialmente com relação à locomoção e à comunicação. O segundo local escolhido foi a Subestação Bateias (SE BTA), localizada em Campo Largo, Paraná, devido à sua importância na transmissão de energia elétrica para Curitiba e Região Metropolitana. Essa subestação está em área rural, levando a maiores dificuldades de comunicação. A Figura 20 mostra o ambiente dessa subestação. Ambas as subestações são operadas pela Copel Geração e Transmissão S.A., que viabilizou e deu suporte para todas as atividades executadas em campo.

Após visitas às subestações, definiu-se, junto à concessionária, as chaves a contemplar nesta pesquisa. Esta escolha teve impacto direto sobre quais técnicas poderiam ser empregadas para identificar o seu estado — por exemplo, uma seleção que fosse restrita a um pequeno conjunto de chaves de um único tipo poderia permitir o uso de técnicas mais simples para o reconhecimento de imagens, mas tal solução não seria generalizável para outras situações. Desta forma, optou-se por um conjunto variado de chaves de diferentes modelos, sendo que todas deverão ser analisadas dentro de um mesmo *framework* de processamento. Também se considerou o fator utilitário para a concessionária, já que há maior importância em ter a informação de estado de circuitos completos ao invés de chaves isoladas. Assim, na SE BTA, foram escolhidas todas as chaves de 525 kV referentes apenas ao circuito denominado “Areia”, cuja região retangular é de aproximadamente 260 metros de extensão por 27 metros de largura.

Figura 19 – Visão geral da Subestação Campo Comprido.
Fonte: Autoria própria (2024).

Já na SE CCO, foram selecionados todos os circuitos do setor de 230 kV, devido à grande diversidade de chaves e à menor extensão dos circuitos. A região selecionada tem extensão de aproximadamente 50 metros e largura de 130 metros.

Na Tabela 2, consta a quantidade de chaves em cada uma das áreas selecionadas, de acordo com o seu tipo. Alguns exemplos de chaves de cada tipo foram previamente mostrados na Figura 1. É importante mencionar que há chaves que, mesmo classificadas em um mesmo tipo, têm aspectos mecânicos e visuais distintos, por diferenças de fabricante, época de fabricação ou outros fatores, conforme ilustrado na Figura 21.

Tabela 2 – Número de chaves nas regiões escolhidas para o estudo, classificadas por tipo.

Subestação	Tipo de chave	Conjuntos trifásicos	Número de chaves
Campo Comprido	Abertura central	18	54
Campo Comprido	Dupla abertura	16	48
Campo Comprido	Semi-pantográfica vertical	2	6
Bateias	Semi-pantográfica horizontal	7	21
Total		43	129

Fonte: Autoria própria (2024).

Figura 20 – Visão geral da Subestação Bateias.
Fonte: Autoria própria (2024).

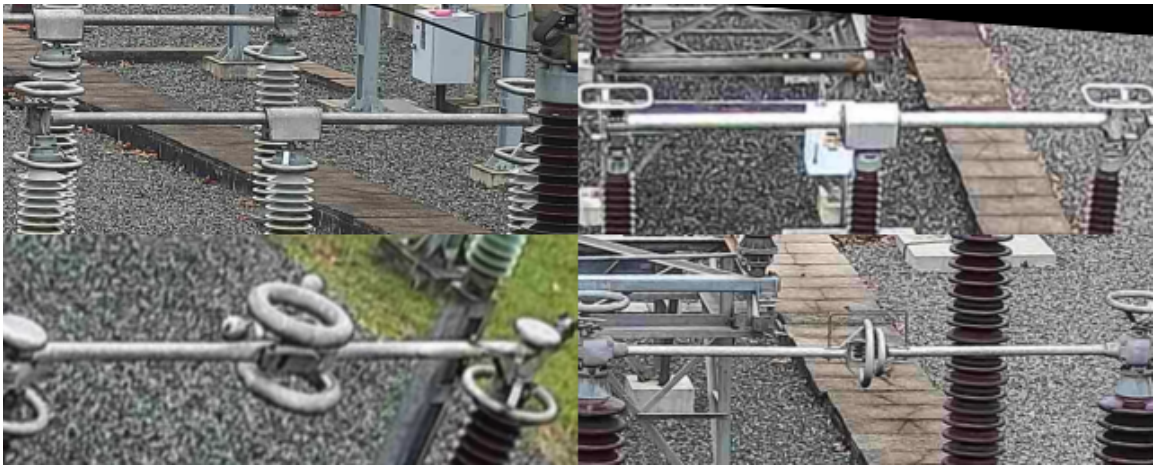


Figura 21 – Diferença de aspecto entre variantes de chaves do mesmo tipo. Acima: dupla abertura; abaixo: abertura central.
Fonte: Autoria própria (2024).

4.2 Modelos tridimensionais das subestações

A fim de possibilitar a geração de dados sintéticos, foram confeccionadas versões virtuais das subestações e seus equipamentos na forma de modelos tridimensionais paramétricos. A atividade de mapeamento das subestações e criação dos modelos tridimensionais foi reali-

zada pela equipe do Lactec, mas ressalta-se que a manipulação posterior desses modelos para a geração de imagens foi de responsabilidade do autor do presente trabalho.

Para auxiliar nessa tarefa, realizou-se o mapeamento de ambas as subestações com a tecnologia LiDAR por meio de *scanners* terrestres, produzindo nuvens de pontos colorizadas de todas as estruturas opacas. A reconstrução virtual das subestações foi feita por modelagem manual, utilizando as nuvens de pontos como alicerce para o desenho de formas tridimensionais. Para atender ao requisito de detalhamento e fidelidade, além das nuvens de pontos, foram utilizadas fotografias dos equipamentos para refinar a modelagem e a coloração dos objetos. No caso da existência de múltiplas instâncias de um mesmo equipamento na subestação (por exemplo, chaves seccionadoras do mesmo tipo e modelo), pôde-se realizar apenas uma modelagem do equipamento, replicando-a conforme a necessidade.

Essas atividades são bastante dispendiosas, principalmente pelo custo de aluguel dos equipamentos de mapeamento e pelo caráter manual da tarefa de modelagem, mas foram consideradas necessárias para o projeto, já que as técnicas de modelagem automática não conseguiriam produzir modelos 3D coerentes no detalhamento de objetos relativamente pequenos em uma área do tamanho de uma subestação. Outro fator que justificou a construção dos modelos 3D (além da finalidade principal, a geração de dados sintéticos) foi a possibilidade de utilizá-los como referência para a definição de pontos estratégicos para o posicionamento das câmeras. A complexidade do cenário dificulta que um humano, a partir do solo, tenha uma noção clara do campo de visão que uma câmera teria em um ponto mais elevado. Com o modelo 3D, pode-se observar um grande número de pontos candidatos em um tempo reduzido, estimando-se virtualmente o ponto de vista de cada câmera, incluindo oclusões e o ângulo sob o qual cada chave seria observada.

Os modelos 3D das subestações foram disponibilizados no formato do *software* Blender, o qual conta com ferramentas de automação para a manipulação da cena virtual e renderização via código-fonte. As visões gerais dos modelos 3D das subestações são apresentadas nas Figuras 22 e 23.

A principal motivação do uso do modelo 3D para a solução deste problema é a capacidade de obter imagens sintéticas das chaves seccionadoras no estado oposto ao que estão normalmente na subestação. Por isso, além da modelagem das chaves, fez-se a animação do seu movimento de abertura, facilitando a manipulação do objeto e possibilitando a geração de imagens da chave em estados intermediários, conforme ilustrado na Figura 24.

Para obter imagens fotorrealistas, o trabalho de modelagem incluiu também a reprodução das características da superfície dos objetos, ou em outras palavras, a definição de materiais, que determinam a forma com que o objeto interage com a luz no processo de renderização. Isso inclui propriedades como cor, textura e refletividade. Além disso, as fontes de luz do modelo 3D podem ser manipuladas quanto à posição, direção, intensidade e cor, permitindo simular a mudança de direção da luz solar ao longo do dia e os refletores para iluminação noturna, como exemplificado na Figura 25.

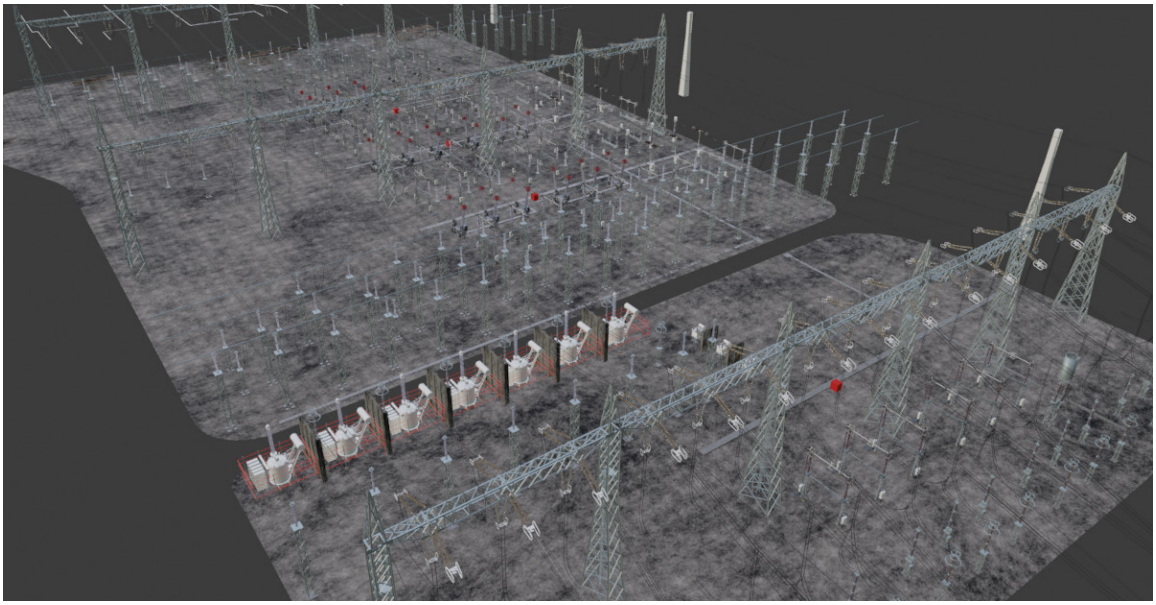


Figura 22 – Visão geral do modelo 3D da Subestação Bateias.
Fonte: Autoria própria (2024).

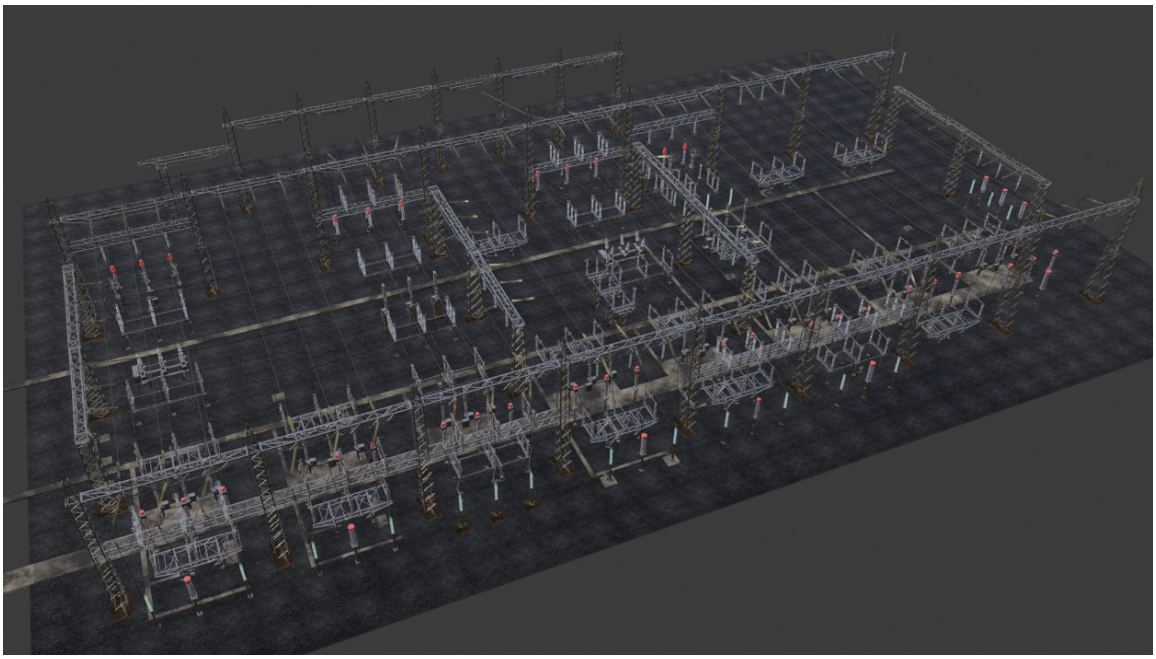


Figura 23 – Visão geral do modelo 3D da Subestação Campo Comprido.
Fonte: Autoria própria (2024).

Outra característica importante para a geração de amostras de treinamento variadas é a possibilidade de reposicionar a câmera virtual, produzindo assim imagens de pontos de vista diferentes. Com isso, pode-se simular qualquer enquadramento produzido por uma câmera em um ponto arbitrário da subestação. Alguns exemplos são ilustrados na Figura 26.

Figura 24 – Animação do fechamento de uma chave seccionadora virtual, em vários estágios do movimento.

Fonte: Autoria própria (2024).

Figura 25 – Efeito da mudança de iluminação sobre uma chave seccionadora virtual.

Fonte: Autoria própria (2024).

Figura 26 – Efeito da mudança de posicionamento da câmera sobre o enquadramento resultante.

Fonte: Autoria própria (2024).

4.3 Projeto e instalação do sistema de captura de imagens

Conforme comentado na seção 4.2, os modelos tridimensionais das subestações tiveram uma finalidade adicional: permitir a definição do posicionamento das câmeras nas subestações, visando obter os melhores enquadramentos de imagens de chaves possíveis com a menor quantidade de equipamentos. Para maximizar a área coberta por cada câmera, deu-se preferência por câmeras do tipo PTZ (*pan*, *tilt*, *zoom*), as quais possuem um sistema motorizado de movimento das lentes, permitindo que uma única câmera monitore diversas chaves seccionadoras em locais diferentes.

A escolha do modelo de câmera foi realizada pela equipe do Lactec, com base em critérios como resolução, capacidade de *zoom* óptico, tamanho do sensor e disponibilidade de uma interface de programação de aplicação (API) para automação da captura, além de fatores como as dimensões das regiões e dos objetos de interesse e a relação custo-benefício do equipamento. Entre as opções disponíveis, a câmera Unitronix UTX6852SR-X38UG se mostrou adequada para a aplicação proposta, a qual conta com resolução de 1920 x 1080 *pixels*, ampliação ótica máxima de 38 vezes e sensor de tipo 1/1.8".

O processo completo de escolha de pontos de instalação nas subestações por meio do modelo 3D é descrito na publicação de Lippmann Jr *et al.* (2022). A simulação fiel de uma câmera específica no modelo 3D requer a configuração dos seus parâmetros ópticos no *software* de modelagem. Para isso, é necessário consultar a ficha técnica do equipamento que se deseja instalar, sendo que as características mais relevantes são (LIPPMANN JR *et al.*, 2022):

- Resolução: largura e altura da imagem capturada, em *pixels*;
- Ângulo de visão (ou campo de visão): extensão angular da cena capturada por uma câmera, em graus. Câmeras com funcionalidade de *zoom* óptico permitem variar o ângulo de visão entre um limite máximo e mínimo. Quanto menor o ângulo de visão, maiores aparecem os objetos.
- Distância focal: distância entre o sensor e o centro óptico da lente da câmera, em milímetros. O ângulo de visão e a distância focal são funcionalmente equivalentes e estão matematicamente relacionados — em certos modelos, a ficha técnica de uma câmera apresenta valores de distância focal ao invés de ângulo de visão. Quanto maior a distância focal, menor o ângulo de visão.
- Tamanho do sensor: dimensões do sensor da câmera, responsável por efetivamente detectar os raios de luz e construir a imagem. Idealmente, o tamanho do sensor deveria ser fornecido pelo fabricante em milímetros (largura e altura), o que ocorre em alguns modelos de câmera. Entretanto, é mais comum encontrar essa informação em um formato legado que data da transição entre as câmeras de tubo *vidicon* e os sensores digitais (por exemplo: 1/1.8"), que não representa precisamente a diagonal do sensor. Há tabelas que mapeiam esse formato para as dimensões em milímetros mais usuais, mas para melhores resultados, deve-se solicitar ao fabricante a informação precisa.

Após consultar as propriedades físicas da câmera e inseri-las no *software* de modelagem (no presente projeto, utilizou-se o Blender), é possível posicionar a câmera em qualquer ponto do modelo 3D e observar o enquadramento resultante. O movimento do sistema PTZ pode ser simulado pela manipulação da rotação da câmera virtual, permitindo avaliar, em um ambiente controlado, quais chaves são visíveis pela câmera em um certo posicionamento, incluindo a ocorrência de oclusões por torres, cabos e outras estruturas. No caso de câmeras com controle de *zoom* óptico, o ângulo de visão (ou a distância focal) é variável no intervalo determinado pelos valores máximo e mínimo, o que permite prever se a imagem capturada terá detalhes suficientes, ou se é necessário posicionar a câmera em um ponto mais próximo do objeto.

Assim, a escolha de pontos de instalação foi feita de forma qualitativa pelo autor, que:

1. Escolheu uma posição inicial empiricamente, considerando critérios como a proximidade aos objetos de interesse, a existência de obstáculos entre a câmera e os objetos,

entre outros. Tratando-se de um ambiente de subestação, há a preferência por instalar as câmeras em estruturas pré-existentes, como torres metálicas, pórticos e postes, já que, para colocar novas estruturas, é necessário atender a normas de projeto de subestações e passar por um processo de autorização.

2. Acessou a visão da câmera virtual e analisou os enquadramentos de imagem para monitorar os objetos de interesse, manipulando as rotações da câmera e o *zoom* (ângulo de visão) no modelo 3D.

3. Corrigiu o posicionamento inicial (quanto à altura, rotação e estrutura de fixação) conforme a qualidade dos enquadramentos obtidos.

Ao completar a avaliação dos enquadramentos no modelo 3D, determinou-se que, para monitorar as chaves escolhidas (seção 4.1), seriam necessárias seis câmeras na SE CCO e cinco câmeras na SE BTA, totalizando onze câmeras. Também determinaram-se os pontos ideais de instalação, todos com intervalos de tolerância de altura para facilitar o trabalho de instalação.

Além das câmeras, outros itens também foram necessários para viabilizar a instalação do sistema de captura, tais como eletrodutos, cabeamentos de energia e comunicação, caixas de energia, *switches* e distribuidores internos ópticos. As atividades de compra dos materiais (incluindo as câmeras), elaboração de esboços das obras a executar nas subestações, contratação de empreiteira especializada e comissionamento da infraestrutura foram realizados pela equipe do Lactec. A Figura 27 ilustra uma das câmeras instaladas para esta pesquisa. Na Figura 28, são ilustradas duas comparações entre a imagem prevista usando o modelo 3D e o enquadramento obtido com a câmera real.

4.4 Formação de conjuntos de imagens reais

Com o conjunto de câmeras instalado fisicamente nas subestações, o passo seguinte foi a preparação de conjuntos de imagens reais para a tarefa de aprendizado de máquina. Para isso, foi necessário definir enquadramentos de câmera que mostrem as partes das chaves seccionadoras importantes para o reconhecimento do seu estado. Câmeras do tipo PTZ normalmente permitem armazenar internamente predefinições de parâmetros a fim de reproduzir um enquadramento conhecido, os chamados *presets*. Isso facilita a captura de imagens de forma desassistida por meio de *scripts* de automação, já que, com a simples ativação de um *preset*, a câmera se move automaticamente para a direção desejada.

As chaves seccionadoras se apresentam em grupos de três chaves, uma para cada fase do circuito, dispostas lado-a-lado e próximas umas das outras. Já os conjuntos trifásicos de chaves de circuitos diferentes estão separados entre si por distâncias maiores. Por causa disso, geralmente não é vantajoso criar um enquadramento em que apareça mais de um con-

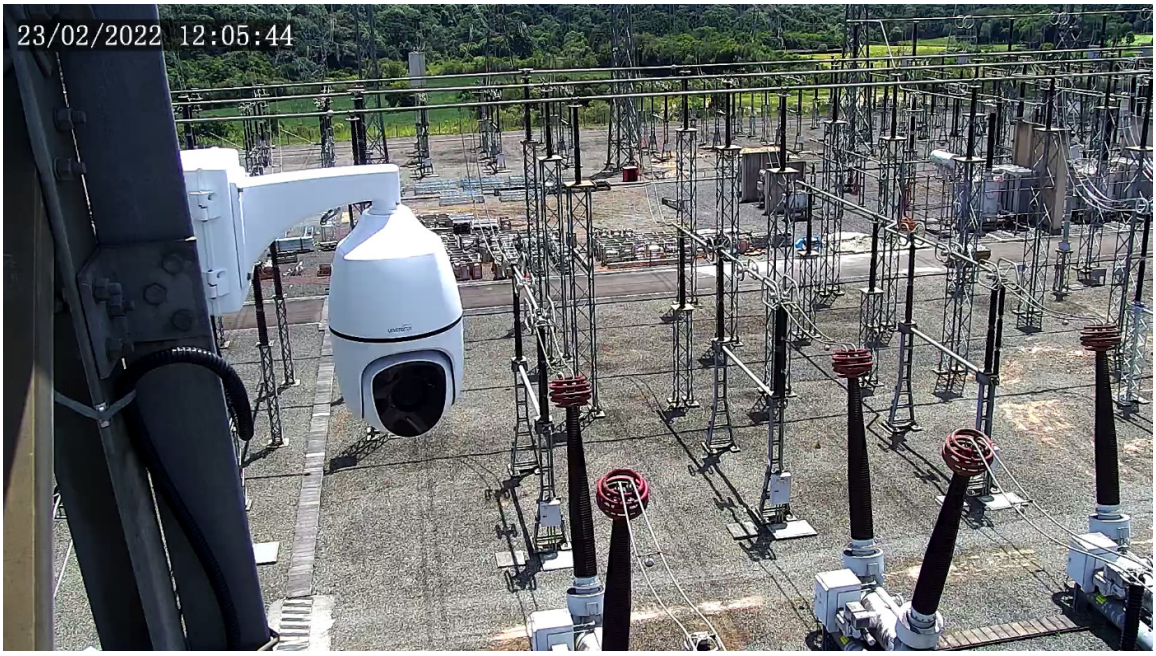


Figura 27 – Uma das cinco câmeras instaladas em uma das torres próximas ao circuito "Areia" na Subestação Bateias.

Fonte: Autoria própria (2024).

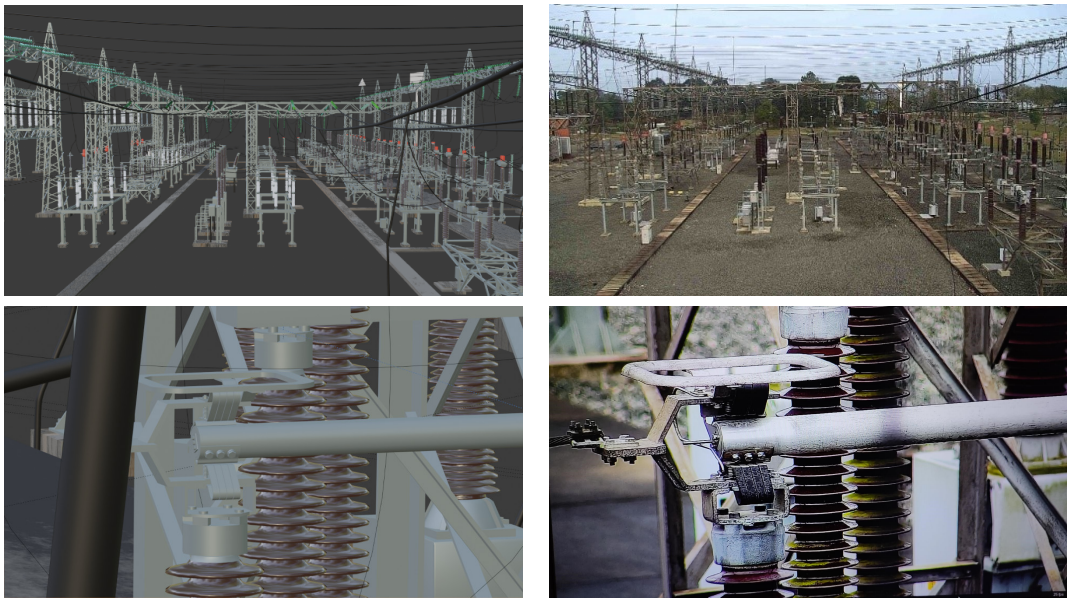


Figura 28 – Comparação entre a visão do *software* de modelagem (à esquerda) e a imagem equivalente capturada pela câmera física (à direita).

Fonte: Adaptado de Lippmann Jr *et al.* (2022).

junto trifásico na mesma imagem, já que para isso seria necessário reduzir o *zoom*, levando a uma diminuição de detalhes. Além disso, as estruturas de suporte que elevam as chaves seccionadoras do nível do chão, como os isoladores e treliças metálicas, são imóveis, o que as torna irrelevantes para o reconhecimento do estado. Assim, o critério principal na definição de cada *preset* foi o enquadramento das três chaves de cada conjunto trifásico específico, focando apenas em suas partes móveis e contatos elétricos.

No total, foram definidos 100 *presets* na Subestação Campo Comprido e 39 presets na Subestação Bateias (exemplificados na Figura 29 e na Figura 30, respectivamente), utilizando a ferramenta de gerenciamento *web* da câmera, que permite fazer essa atividade de forma interativa. É importante lembrar que nem todas as chaves são visíveis a partir de todas as câmeras, por conta da distância, oclusão por outros objetos e outros fatores. Há, também, vários *presets* redundantes, ou seja, que produzem imagens de uma mesma chave a partir do ponto de vista de outra câmera. Estas redundâncias não são problemáticas, e em uma versão final do sistema poderiam ser exploradas para aumentar a robustez do sistema ao permitir múltiplas observações para uma mesma chave — do ponto de vista da classificação como abordado no presente trabalho, a visão de uma mesma chave por duas câmeras é bastante diferente, e cada ponto de vista será analisado de forma independente. Também há alguns casos em que, devido à proximidade das chaves em relação à câmera, foi necessário definir dois presets contendo apenas parte do conjunto, pois não se conseguiu criar um enquadramento das três chaves em uma mesma imagem.

Figura 29 – Exemplos de *presets* definidos para as câmeras instaladas na Subestação Campo Comprido.

Fonte: Autoria própria (2024).

Na sequência, desenvolveu-se um *script* para controle de câmeras e captura automatizada de imagens. Para isso, utilizou-se a API disponibilizada e devidamente documentada pelo fabricante das câmeras, a qual utiliza o protocolo HTTP na forma de *web services*. O *script* exe-

Figura 30 – Exemplos de *presets* definidos para as câmeras instaladas na Subestação Bateias. Fonte: Autoria própria (2024).

cuta varreduras periódicas, passando por todos os presets em cada ciclo e salvando as capturas em memória não-volátil. A Figura 31 mostra algumas imagens capturadas dessa forma, para um *preset* específico. Para reduzir o efeito de ruídos e de elementos que passem diante da câmera (por exemplo, gotas de chuva), cada fotografia é obtida a partir da média *pixel*-a-*pixel* de três capturas seguidas, alinhadas com o algoritmo de registro descrito na seção 4.5. Após o término de uma varredura, a próxima é iniciada apenas após um tempo parametrizado (de cerca de uma hora, na maior parte das capturas realizadas), já que o objetivo principal é providenciar imagens nas mais variadas condições, não apenas de iluminação, mas também climáticas, como chuva, neblina, vento e outras. Também buscou-se capturar variações de longo prazo, provocadas por diferenças de iluminação em meses diferentes.

Mesmo com a execução de varreduras em diversos momentos, esse procedimento, por si só, é apenas capaz de capturar imagens das chaves em seu estado usual (que pode ser aberto ou fechado, a depender de sua função na subestação). Ao longo dos períodos descritos, houve muito poucas mudanças de estado de um pequeno número de chaves, fazendo com que fosse necessário realizar atividades de manobra para complementar a base de dados para fins de teste. Devido à presteza da equipe da Copel Geração e Transmissão, foi possível viabilizar al-

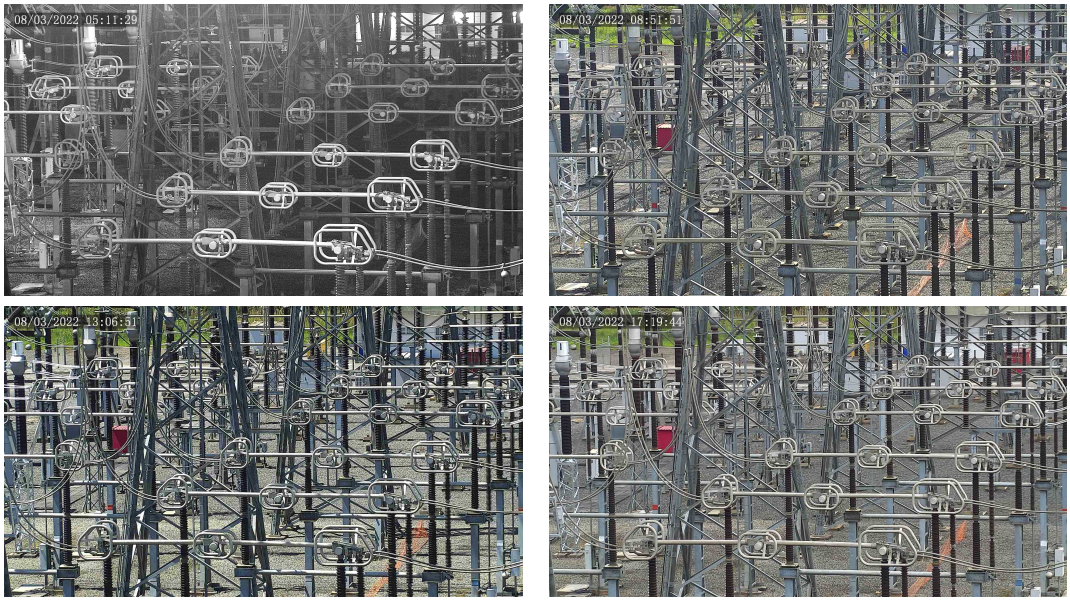


Figura 31 – Capturas realizadas com o processo de varredura ao longo de um mesmo dia, para um enquadramento específico.

Fonte: Autoria própria (2024).

gumas manobras em datas específicas, mediante agendamento prévio, autorização dos órgãos competentes e mobilização de equipe. Um exemplo de manobra é ilustrado na Figura 32.

Figura 32 – Comparação entre o aspecto de um conjunto de chaves em seu estado usual (fechado), à esquerda, com seu estado manobrado (aberto), à direita. As manobras foram realizadas com o objetivo de capturar imagens exclusivamente para teste da solução proposta.

Fonte: Autoria própria (2024).

A Tabela 3 contém informações sobre os conjuntos de imagens reais utilizados nos testes com modelos de aprendizado de máquina. Para as duas subestações, os conjuntos “Normal 1” e “Normal 2” possuem apenas imagens das chaves nos seus estados usuais, sendo que a diferença entre os dois é que o primeiro foi usado para realizar testes e comparações, guiando escolhas realizadas durante o desenvolvimento, enquanto o segundo foi usado exclusivamente para testar os modelos finais produzidos. Já os conjuntos de “Manobras” contêm tanto chaves manobradas quanto no seu estado usual, em capturas realizadas em um mesmo dia. É evidente que há uma desproporcionalidade entre o número de imagens de chaves na situação usual e na situação manobrada, precisamente o que motivou a ideia de utilizar dados sintéticos

no treinamento. Estes três conjuntos de capturas foram reservados exclusivamente para teste dos modelos de inteligência artificial.

Tabela 3 – Capturas de imagens realizadas em ambas as subestações, para testar os modelos de aprendizado de máquina.

Subestação	*Dataset*	Data(s)	Capturas
Campo Comprido	Normal 1	12/03/2022 até 18/03/2022	15.036
	Normal 2	04/06/2022 e 05/06/2022	4.099
	Manobras	02/02/2022 e 15/12/2022	72
Bateias	Normal 1	04/03/2022 até 09/03/2022	1.847
	Normal 2	03/06/2022 até 05/06/2022	1.415
	Manobras	30/08/2022	99

Fonte: Autoria própria (2024).

4.5 Geração de referências e rotulagem de imagens

Antes de reconhecer o estado de cada chave seccionadora, o sistema deve ser capaz de encontrar cada chave na imagem (conforme comentado na seção 4.4, há, fora exceções, três chaves por imagem) e identificá-la de maneira única e individual, conforme a nomenclatura adotada pela concessionária. Fazer essa identificação de maneira automática e baseada somente no aspecto de cada chave seria infactível, e mesmo humanos dependem de marcações e identificadores para realizar a tarefa (ver Figura 33). Entretanto, os identificadores existentes nas subestações foram concebidos e posicionados para serem lidos por humanos no ambiente da subestação, e não aparecem na maior parte dos enquadramentos definidos para monitorar as chaves. Isso poderia ser contornado pelo posicionamento de novos marcadores junto às próprias chaves, mas essa ação não é viável pelas dificuldades de manutenção do sistema (vide seção 1.1).

Para permitir a identificação das chaves sem a necessidade de se adicionar marcadores físicos à subestação, adotou-se a mesma abordagem apresentada por Nassu *et al.* (2022). Para cada *preset* de câmera, um rotulador humano realiza marcações que indicam a região de cada chave, o seu identificador, e outras informações úteis para o reconhecimento. Contudo, as câmeras estão expostas à influência do vento, fazendo com que as imagens capturadas em um mesmo *preset* tenham pequenas variações devido ao balanço, e também à imprecisão dos motores do sistema PTZ, o que torna as marcações imprecisas. Essas variações, quando pequenas, podem ser consideradas simples translações, podendo ser solucionadas por um algoritmo de alinhamento (registro) a uma imagem de referência. Assim, para cada *preset* de câmera, foi gerada uma imagem de referência, sobre a qual o rotulador faz as anotações, e à qual todas as imagens subsequentes capturadas pelo sistema devem ser alinhadas. Deve-se notar que essas operações são realizadas somente uma vez por *preset*, precisando apenas

Figura 33 – Identificador de um conjunto de chaves seccionadoras, posicionado na caixa do seu circuito de comando.

Fonte: Autoria própria (2024).

serem repetidas caso o *preset* mude significativamente, e provêm informações essenciais para o sistema tanto em tempo de treinamento quanto em tempo de inferência.

Para permitir o alinhamento de imagens capturadas em qualquer horário, as imagens de referência devem mostrar as chaves monitoradas sem problemas de visibilidade causados por má iluminação, sombras ou chuva — ou seja, uma imagem com iluminação tão neutra quanto possível, como exemplificado na Figura 34. Para gerar uma imagem de referência de um *preset*, é computada a média *pixel*-a-*pixel* de várias imagens capturadas para o *preset* ao longo de pelo menos um dia inteiro (incluindo o período noturno). Neste trabalho, considerou-se uma quantidade superior a 100 imagens para cada *preset*. Armazena-se a imagem resultante em formato PNG com compressão sem perdas, o que, aliado à média de várias imagens, torna-a menos sujeita à presença de ruído e artefatos de compressão. As imagens utilizadas para construir uma referência devem ser alinhadas umas às outras para que a imagem média tenha boa definição, como mostrado na Figura 35, e para isso também se usa o algoritmo de registro.

O alinhamento entre imagens de um mesmo *preset*, seja para alinhar várias capturas para a produção de uma imagem de referência ou para alinhar uma captura individual a uma referência já existente, é realizado por um algoritmo de registro. Esse é um problema clássico na área de processamento digital de imagens, para o qual já existe um grande número de soluções (ZITOVÁ; FLUSSER, 2003). Um fator importante para a escolha da solução de alinhamento em uma aplicação específica é a caracterização das variações de enquadramento esperadas nas imagens. Há abordagens que modelam transformações complexas, como transformações

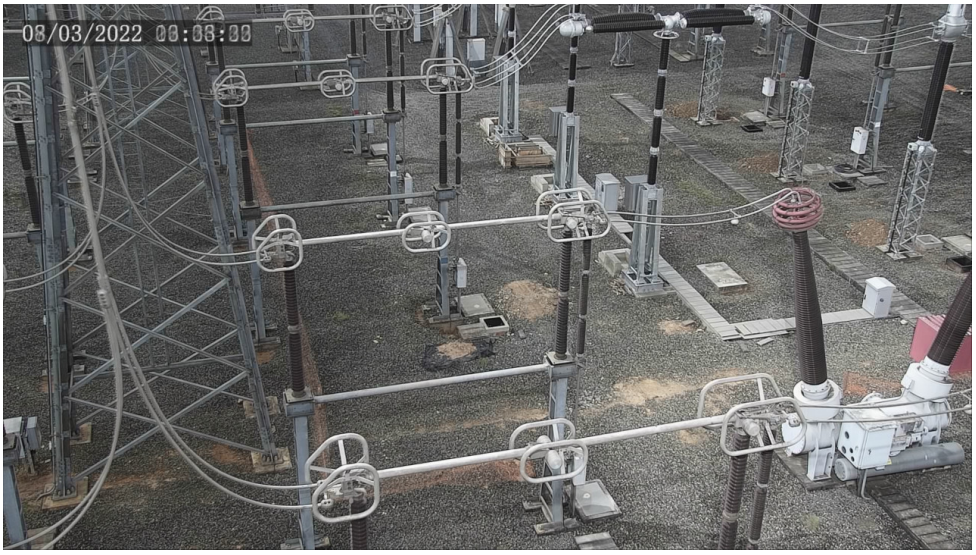


Figura 34 – Imagem de referência produzida a partir da média de várias capturas do mesmo enquadramento alinhadas umas às outras.

Fonte: Autoria própria (2024).

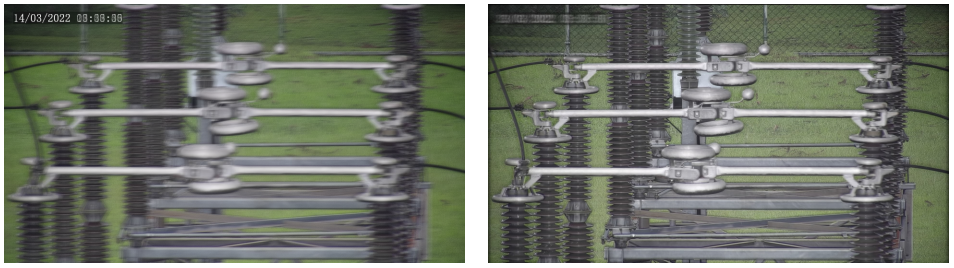


Figura 35 – Média de várias capturas de um mesmo enquadramento ao longo de um dia. Na figura à esquerda, não se utilizou registro; já na figura à direita, sim.

Fonte: Autoria própria (2024).

de escala, rotação e perspectiva planar, mas para o propósito do presente trabalho, uma abordagem que modela apenas translações é suficiente para produzir bons resultados, além de ser computacionalmente mais econômica. Assim, aplicou-se a mesma técnica apresentada por Nassu *et al.* (2022): uma variação da abordagem clássica baseada em correlação de fase no domínio da frequência (FOROOSH; ZERUBIA; BERTHOD, 2002) aplicada sobre o mapa de magnitudes dos gradientes das imagens para atenuar dificuldades causadas pelas variações de iluminação.

Uma vez tendo computadas as imagens de referência, um especialista humano deve fazer a rotulagem dos *presets*, que consiste em indicar a localização e o identificador de cada chave nas imagens, além de informar se a chave está aberta ou fechada. Para facilitar essa tarefa, desenvolveu-se um programa que permite que o rotulador insira as anotações de forma interativa, sobre a própria imagem. O programa foi escrito na linguagem Python[1], com o uso de

[1] https://www.python.org/

bibliotecas como NumPy[2] e OpenCV[3], e as anotações referentes a cada *preset* são armazenadas em arquivos de texto.

O procedimento para marcação de chaves é ilustrado na Figura 36 e na Figura 37. Para cada chave seccionadora presente na imagem, o rotulador deve informar o identificador (código pelo qual a chave é identificada pela concessionária), o modelo (por exemplo, dupla abertura, abertura central, etc.) e dois pontos nas extremidades da chave. A marcação de dois pontos é suficiente porque há uma característica comum a todas as chaves seccionadoras das subestações estudadas, independentemente do tipo ou modelo: quando fechadas, o dispositivo mecânico permanece retilíneo, conectando eletricamente dois pontos. Assim, os dois pontos podem ser utilizados tanto para delimitar a região em que a chave aparece na imagem quanto para indicar a inclinação do corpo alongado quando a chave se encontra fechada.

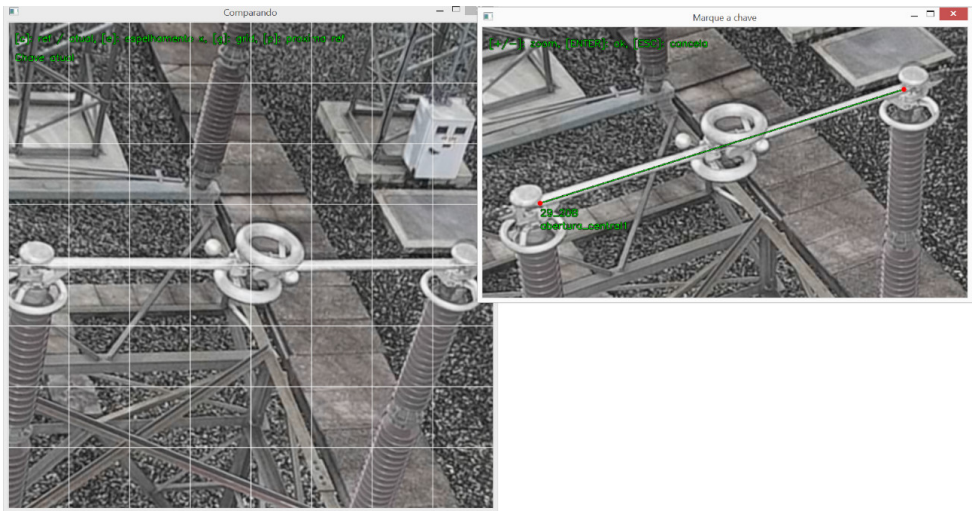


Figura 36 – Procedimento de marcação de chaves. Após informar o identificador da chave, o usuário marca dois pontos nas extremidades da chave utilizando a tela à direita. Na tela à esquerda, mostra-se a visualização da imagem transformada, na qual os dois pontos são movidos para locais fixos, deixando a chave na horizontal. A visualização é atualizada dinamicamente, conforme o usuário move os pontos marcados.

Fonte: Autoria própria (2024).

Apesar do trabalho de preparação dos enquadramentos, nem todas as (até) três chaves de cada enquadramento têm boa visibilidade, pois podem estar obstruídas por estruturas da subestação ou estar em um ponto de vista desfavorável. Por isso, durante a rotulagem, foram marcadas em cada imagem apenas as chaves que não são significativamente afetadas por esses problemas. A Tabela 4 mostra a quantidade total de chaves marcadas entre todos os enquadramentos, que representa o número de instâncias do problema de reconhecimento quanto aos diferentes pontos de vista.

[2] https://numpy.org/
[3] https://opencv.org/

Figura 37 – Imagem após o procedimento de marcação, com as chaves rotuladas em evidência.
Fonte: Autoria própria (2024).

Tabela 4 – Pontos de vista marcados para cada tipo de chave.

Subestação	Tipo de chave	Pontos de vista
Campo Comprido	Abertura central	146
Campo Comprido	Dupla abertura	119
Campo Comprido	Semi-pantográfica vertical	15
Bateias	Semi-pantográfica horizontal	93
Total		373

Fonte: Autoria própria (2024).

4.6 Correspondência entre o ambiente real e o virtual

Para promover a transferência de aprendizado do domínio dos dados sintéticos gerados por renderização para o domínio real, os enquadramentos definidos para as câmeras físicas instaladas nas subestações (seção 4.4) foram reproduzidos nos modelos 3D. Em outras palavras, isso representa uma nova definição de *presets*, mas agora no ambiente virtual, usando como base as imagens de referência produzidas com capturas reais (seção 4.5).

Este trabalho consistiu em determinar, para cada enquadramento, a sétupla $(p_x,p_y,p_z,r_x,r_y,r_z,\alpha)$ que gera o enquadramento semanticamente equivalente no ambiente 3D, tal que p_x, p_y e p_z são as coordenadas cartesianas de posição da câmera, r_x, r_y e r_z indicam a rotação do sistema de coordenadas da câmera em torno de cada eixo cartesiano (ângulos de Euler) e α representa o ângulo de visão da câmera, que dá *zoom* à imagem. Isso é realizado por meio da manipulação manual dos sete parâmetros da câmera virtual no *software* de modelagem, alinhando a imagem produzida na simulação com a imagem real de referência.

À primeira vista, pode-se pensar que os sete graus de liberdade são mais que o necessário, pois a câmera fica instalada em uma posição fixa, e um enquadramento da câmera PTZ

é definido por apenas três parâmetros: dois de rotação e um valor de *zoom*. Porém, na prática, observou-se que essa simplificação não gera uma boa correspondência entre o mundo real e o virtual, pois:

- A câmera real é um corpo extenso cujo centro de rotação (do mecanismo PTZ) não coincide com o seu centro óptico, resultando em uma translação, embora pequena, do centro óptico quando é rotacionada. Para reproduzir esse desvio com a câmera virtual, são necessários os três parâmetros de posição.
- A superfície onde a câmera é instalada pode não estar paralela ao eixo vertical, pois torres e postes têm base mais larga que o topo por questão de estabilidade mecânica. Por isso, a câmera pode ter uma inclinação axial, o que leva à necessidade de um parâmetro adicional de rotação na câmera virtual.

Para auxiliar nessa tarefa, há uma ferramenta no *software* de modelagem Blender que permite adicionar uma imagem de fundo ao *preview* 3D. Com isso, é possível observar o modelo 3D sobreposto a uma imagem real do referido preset, permitindo movimentar a câmera e melhorar o alinhamento entre os objetos virtuais e os reais. Um exemplo desse processo está ilustrado a seguir. A Figura 38 é a imagem de referência de um preset (média de várias capturas reais após registro, conforme explicado na seção 4.5). Na Figura 39, a imagem de referência foi colocada sob a visualização virtual do mesmo preset, comprovando que, embora as imagens estivessem parecidas, havia um pequeno desalinhamento. Em especial, notou-se que a câmera real não estava perfeitamente paralela ao eixo vertical, provavelmente devido à angulação dos perfis metálicos da torre. Após as devidas correções no preset virtual, gerou-se a Figura 40, na qual o enquadramento virtual se apresenta mais coerente ao real.

Figura 38 – Exemplo de imagem real de referência utilizada para a definição de um *preset* virtual. Fonte: Autoria própria (2024).

É importante lembrar que o ambiente 3D é uma aproximação da realidade, e por isso não é possível produzir um alinhamento perfeito, já que existem imprecisões e não linearidades no comportamento da câmera física que não são facilmente simulados no ambiente virtual. Mesmo assim, basta que ambas as versões de um enquadramento (real e virtual) sejam semanticamente equivalentes, ou seja, que os objetos relevantes apareçam aproximadamente nas mesmas posições.

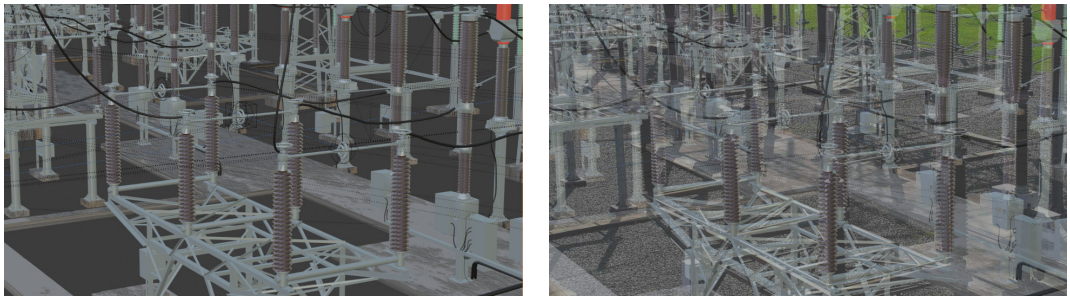


Figura 39 – Imagem virtual equivalente ao *preset* da Figura 38, mas sem considerar as correções de posicionamento e rotação. À direita, mostra-se a imagem virtual sobreposta à real, na qual o efeito "fantasma" evidencia um desalinhamento não desprezível.
Fonte: Autoria própria (2024).

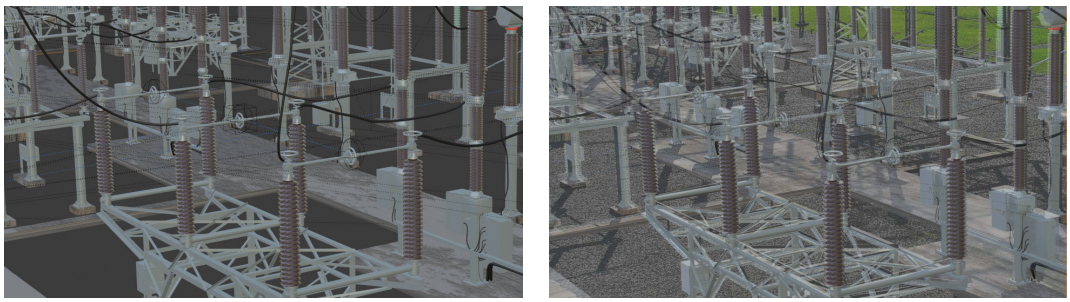


Figura 40 – Idem à Figura 39, mas após as correções de posicionamento e rotação. A imagem virtual tornou-se muito mais próxima à real, especialmente na região das chaves. Ainda é possível observar um ligeiro desalinhamento nos objetos mais distantes da câmera, mas não há vantagem prática em melhorá-lo além deste ponto.
Fonte: Autoria própria (2024).

Após a definição dos *presets* virtuais, fez-se a renderização de uma imagem de cada *preset* com condições neutras de iluminação, produzindo, assim, imagens de referência do modelo 3D. Alguns exemplos são mostrados na Figura 41. Apesar de os enquadramentos virtuais terem sido definidos à semelhança dos reais, realizou-se uma nova rotulagem para as imagens sintéticas utilizando o mesmo procedimento descrito na seção 4.5, já que os pontos de referência das chaves nas imagens reais e sintéticas podem não coincidir perfeitamente. Com isso, garante-se a capacidade de o sistema localizar as chaves seccionadoras tanto nas imagens reais quanto nas imagens sintéticas.

4.7 Geração de imagens a partir do modelo 3D

Com a descrição de todos os *presets* virtuais como sétuplas de parâmetros de câmera, é possível renderizar imagens a partir do modelo 3D enquanto se faz a manipulação dos seus parâmetros, introduzindo uma abordagem de randomização de domínio (ver Seção 2.1.4). Entretanto, para alcançar a quantidade necessária de imagens sintéticas, modificando os atributos da simulação a cada imagem gerada, deve-se automatizar o processo de geração de alguma forma.

Figura 41 – Exemplos de resultados do procedimento de correspondência entre *presets* virtuais e reais.

Fonte: Autoria própria (2024).

O *software* Blender, no qual os modelos 3D foram produzidos, disponibiliza uma interface de programação de aplicação (API) na linguagem Python com recursos que possibilitam a manipulação da cena 3D e a renderização de forma desassistida. Assim, elaborou-se um *script* que consiste, basicamente, em executar um conjunto de operações para cada imagem a renderizar:

1. Configurar a posição, rotação e *zoom* da câmera conforme o *preset* virtual que foi selecionado;
2. Manipular, de forma aleatória, as chaves seccionadoras referentes ao *preset* entre as posições aberta e fechada;
3. Manipular, de forma aleatória, as demais chaves seccionadoras (que não são monitoradas pelo *preset*, mas podem aparecer ao fundo no enquadramento), deixando-as em qualquer posição (inclusive estados intermediários ou entreabertos);
4. Manipular, de forma aleatória, os parâmetros do ambiente;
5. Renderizar uma imagem com a câmera virtual, salvando o resultado como um arquivo PNG;
6. Salvar a informação do estado aberto ou fechado das chaves seccionadoras referentes ao *preset*, pois é utilizada durante o treinamento.

Durante a configuração da posição da câmera no modelo 3D, aplica-se uma pequena perturbação aleatória em sua posição e rotação, de modo a simular a influência do vento observada na câmera real instalada em campo. Essa perturbação em terceira dimensão produz

variações de enquadramento significativas devido à vista em perspectiva e ao efeito de paralaxe, ou seja, a sensação de que, ao mover a câmera, os objetos que estão mais próximos aparentam se mover mais no enquadramento do que os mais distantes. Alguns exemplos são mostrados na Figura 42. Para manter as chaves sobre os pontos definidos na rotulagem, as imagens geradas após a perturbação da câmera são realinhadas usando o mesmo algoritmo de registro descrito na seção 4.5, usando a imagem sintética de referência do respectivo *preset* (gerada como descrito na seção 4.6).

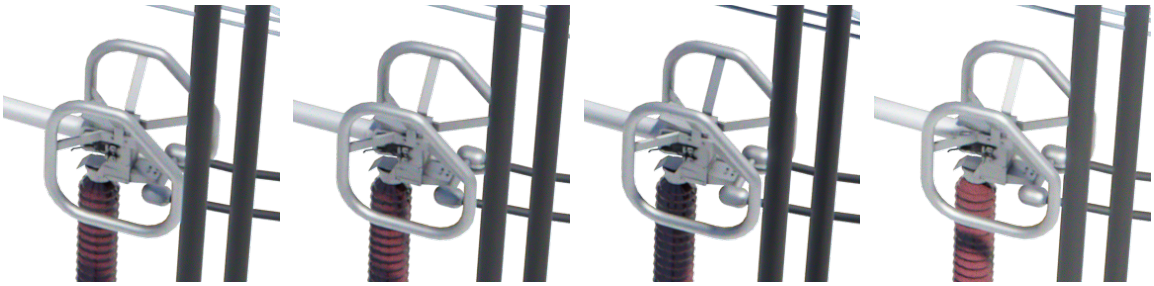


Figura 42 – Recortes de renderizações para ilustrar o efeito da perturbação aleatória da câmera. Comparando as imagens, é possível perceber que o enquadramento varia levemente, mais evidente pelo movimento relativo entre o cabo e a chave (paralaxe).
Fonte: Autoria própria (2024).

A animação de abertura de cada chave no modelo 3D é controlada por um único parâmetro: o grau de abertura, variável no intervalo de 0,0 a 1,0, em que 0,0 é totalmente fechado e 1,0 é totalmente aberto. Entretanto, na geração de imagens, considerou-se um intervalo de tolerância de grau de abertura em ambos os estados, pois as chaves físicas na subestação nem sempre chegam totalmente no final de seu curso ao serem manobradas. Isso também contribui para uma ligeira mudança de aspecto da chave na imagem sintética, contribuindo para a randomização de domínio. No presente trabalho, as margens de tolerância do grau de abertura foram definidas em 0,02 para ambos os estados, ou seja, imagens de chaves fechadas são geradas aleatoriamente entre 0,00 e 0,02, e imagens de chaves abertas entre 0,98 e 1,00. Na Figura 43, mostra-se uma comparação entre imagens geradas nos dois limites do intervalo de tolerância para chaves no estado fechado. O *script* foi codificado de forma que, em cada enquadramento, cada chave tenha um número igual de observações no estado aberto e no estado fechado, visando manter o equilíbrio de classes no conjunto de dados produzido. Em alguns enquadramentos, podem também aparecer chaves de outros conjuntos trifásicos ao fundo — essas chaves são aleatoriamente manipuladas para qualquer abertura entre 0,0 e 1,0, para que o aprendizado não as considere na decisão do estado das chaves pertinentes ao enquadramento em questão.

Outro parâmetro importante que é variado a cada imagem gerada é o controle da iluminação. Para simular a luz solar das imagens diurnas, utilizou-se o modelo de iluminação direcional na cena virtual, no qual os raios de luz são gerados em uma única direção predefinida, como se a fonte de luz estivesse no infinito. No modelo 3D criado para este trabalho, a direção da iluminação foi modelada conforme o movimento relativo do Sol no céu durante o dia, ou seja,

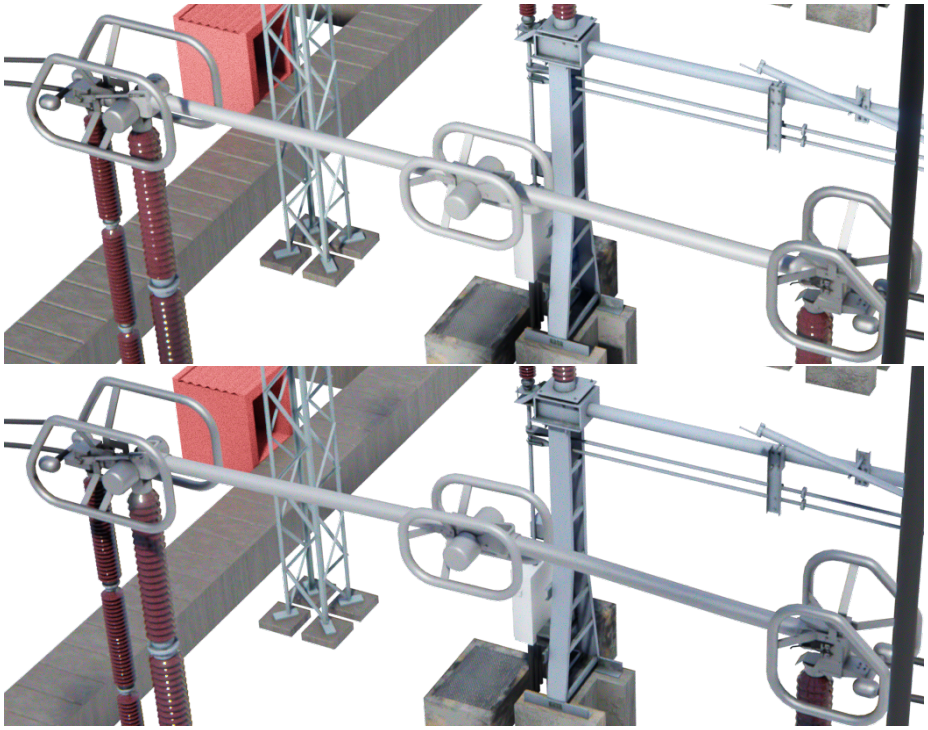


Figura 43 – Renderizações mostrando a tolerância do grau de abertura para o estado fechado. Acima, a chave está plenamente fechada (grau de abertura igual a 0,0). Abaixo, a chave está no limite de tolerância para o estado fechado (grau de abertura igual a 0,02).
Fonte: Autoria própria (2024).

o horário, considerando-se a localização geográfica das duas subestações. Assim, pode-se gerar imagens sintéticas com padrões de sombra distintos, equivalentes à passagem das horas. Também foi introduzido um parâmetro para reproduzir a inclinação dos raios solares, que ocorre a depender da latitude do local e também ao longo do ano, permitindo simular imagens, por exemplo, em época de inverno, em que os raios de luz chegam com uma maior inclinação. Um exemplo é ilustrado na Figura 44. Adicionalmente, a cor e intensidade da luz também podem ser modificadas, permitindo, por exemplo, criar o tom avermelhado característico da alvorada e crepúsculo, ou então reproduzir a iluminação mais fraca de um dia nublado. O modelo 3D também permite gerar imagens sintéticas simulando a iluminação artificial produzida por refletores ligados à noite. Entretanto, as câmeras utilizadas ativam um modo de captura noturna no espectro infravermelho próximo (*near-infrared*) nestas situações — esse tipo de captura não foi simulado no modelo 3D.

Outra característica disponível no ambiente 3D é a geração procedural do chão de brita por meio de padrões aleatórios de ruído. O resultado é uma superfície rugosa em três dimensões que imita a face das pequenas pedras, interagindo, assim, com a iluminação e produzindo sombras mais realistas, conforme ilustrado na Figura 44. Mesmo com esse recurso disponível, observou-se que o chão de brita não aparece em todos os enquadramentos das câmeras reais, a depender da localização da chave e da direção do enquadramento. Em alguns casos, aparecem outros cenários ao fundo, como gramado, asfalto, edificações, árvores, céu, entre outros.

Por causa disso, no *script*, a geração de brita na imagem sintética é habilitada aleatoriamente com uma chance parametrizável.

Figura 44 – Renderizações com chão de brita gerado de forma procedural e com variações de iluminação. Nota-se que a direção das sombras é diferente nas duas imagens.
Fonte: Autoria própria (2024).

Com relação aos diferentes cenários de fundo encontrados nas imagens reais, também buscou-se obter variedade nas imagens sintéticas. Contudo, isso não foi feito em tempo de renderização, pois não seria possível modelar uma quantidade exaustiva de possíveis objetos de fundo. Em vez disso, a renderização foi configurada para produzir imagens com fundo transparente nas áreas sem nenhum objeto, e, durante a extração de amostras (seção 4.8), aplicou-se a técnica de composição 2D com imagens de fundo aleatórias. Dessa forma, cada imagem renderizada pôde ser combinada com diferentes imagens de fundo, gerando uma quantidade ainda maior de amostras de treinamento. Com isso, otimizou-se o tempo para produção de um conjunto de dados, já que a renderização 3D é computacionalmente mais custosa que a composição 2D.

4.8 Extração de amostras para aprendizado de máquina e inferência

Para viabilizar o reconhecimento do estado das chaves seccionadoras, as imagens de entrada dos modelos de *machine learning* não devem ser diretamente as fotografias capturadas pelas câmeras (ou as renderizações equivalentes), pois essas imagens são grandes e possuem detalhes irrelevantes, além de poderem mostrar mais de uma chave. Dessa forma, as amostras utilizadas para treinamento dos modelos, assim como as imagens sobre as quais são realizadas inferências, são recortes menores que mostram somente uma das chaves em posição centralizada.

Um caso de extração de amostras é mostrado na Figura 45. A captura original mostra diversos elementos irrelevantes além das duas chaves que são o alvo principal deste *preset*. Dessa forma, são extraídas duas amostras desta fotografia, cada uma mostrando uma das chaves centralizada, rotacionada para que a sua estrutura apareça horizontalmente quando a chave está fechada, e com o tamanho normalizado. Essas transformações geométricas são baseadas na marcação realizada para cada *preset*, explicada na seção 4.5. A amostra final possui 320 x 128 *pixels*, tamanho escolhido com o objetivo de equilibrar desempenho e custo computacional.

Figura 45 – Extração de amostras (abaixo) de uma captura de câmera (acima).
Fonte: Autoria própria (2024).

O procedimento também é realizado para as imagens geradas a partir do modelo tridimensional, mas há um passo adicional: a composição 2D com uma imagem de fundo aleatória. O fato de que as renderizações possuem fundo transparente e *anti-aliasing* (*pixels* translúcidos na transição entre os objetos e o fundo) permite uma composição mais natural. Seleciona-se

uma região aleatória de uma imagem aleatória, em um conjunto de 400 fotografias de fundo que inclui localidades diversas (Figura 46), a maioria mostrando ambientes e elementos de subestações de transmissão ou distribuição de energia elétrica. A região de fundo selecionada pode sofrer uma série de transformações, como redimensionamento, borramento, rotação, e mudanças de brilho e contraste.

Figura 46 – Algumas das imagens que fazem parte do conjunto de planos de fundo adicionados às renderizações por composição 2D

Fonte: Autoria própria (2024).

As amostras geradas podem apresentar situações pouco realistas, mas isso não é considerado um problema, visto que o objetivo da adição destes cenários de fundo não é ilustrar situações reais, mas adicionar às amostras uma grande variedade de elementos que poderiam levar a confusão e dificuldades para sistemas de reconhecimento automático, como estruturas alongadas e retas em certos ângulos e posições. A amostra após composição pode, ainda, sofrer um leve borramento, para simular variações de foco, além de sobreposição de um padrão de iluminação que modifica o equilíbrio das cores, e a adição de ruído gaussiano. Alguns exemplos estão ilustrados na Figura 47.

Figura 47 – Amostras extraídas de renderizações depois do pós-processamento.

Fonte: Autoria própria (2024).

Pode-se argumentar que a decisão de extrair amostras com as chaves centralizadas pode levar a um viés dos modelos treinados a reconhecer padrões de chave apenas nas regiões centrais da imagem. Entretanto, o processo descrito assegura que todas as imagens apresentadas ao modelo, seja em tempo de treinamento ou inferência, têm essa característica, configurando, portanto, uma mera delimitação de domínio.

4.9 Modelos de redes convolucionais

O reconhecimento do estado das chaves nas imagens capturadas foi tratado como um problema de classificação, cuja entrada é uma imagem com 320 x 128 *pixels* que mostra apenas uma chave em primeiro plano, e a saída é o estado identificado: fechada ou aberta. Para resolver o problema, foram adotadas soluções baseadas em redes neurais convolucionais. Neste primeiro momento, o objetivo dos experimentos foi investigar o potencial do uso de dados sintéticos para a aplicação proposta, sem avaliar comparativamente a influência de cada técnica, arquitetura ou escolha de parâmetros.

O desenvolvimento inicial da solução foi feito com uma arquitetura tradicional simples (Tabela 5), composta por camadas intercaladas de convolução 2D, ativação ReLU e *max pooling*, seguida por um classificador *multilayer perceptron* com a função *softmax* na saída. Também foi elaborada uma versão expandida dessa arquitetura (Tabela 6), modificando a largura das camadas.

Tabela 5 – Arquitetura da rede convolucional simples utilizada nos testes iniciais.

Entrada	Operação	Parâmetros treináveis
$128 \times 320 \times 3$	conv 3x3 + relu + pool 2x2	224
$64 \times 160 \times 8$	conv 3x3 + relu + pool 2x2	1.168
$32 \times 80 \times 16$	conv 3x3 + relu + pool 2x2	4.640
$16 \times 40 \times 32$	fc + relu	655.392
$1 \times 1 \times 32$	dropout 20%	0
$1 \times 1 \times 32$	fc + softmax	66
$1 \times 1 \times 2$	saída	Total: 661.490

Fonte: Autoria própria (2024).

Outra construção de rede neural avaliada está ilustrada na Tabela 7. Essa rede tem o objetivo de manter a maior capacidade de aprendizado nas camadas convolucionais, que conseguem explorar as relações espaciais entre os valores de entradas, ao invés de manter a maior quantidade de parâmetros nas camadas totalmente conectadas ao final da rede, como ocorre na rede simples. Para isso, as operações de *pooling* são substituídas por convoluções com *stride* 2x2, como proposto por Springenberg *et al.* (2015). Além disso, substituiu-se uma das camadas totalmente conectadas pela operação *Global Average Pooling* (GAP), proposta por Lin, Chen e Yan (2014), que transforma o tensor de saída das camadas convolucionais em um vetor unidimensional por meio da média global da ativação de cada *feature map*, diferentemente da

Tabela 6 – Arquitetura da rede convolucional simples expandida.

Entrada	Operação	Parâmetros treináveis
$128 \times 320 \times 3$	conv 3x3 + relu + pool 2x2	896
$64 \times 160 \times 32$	conv 3x3 + relu + pool 2x2	18.496
$32 \times 80 \times 64$	conv 3x3 + relu + pool 2x2	73.856
$16 \times 40 \times 128$	fc + relu	10.485.888
$1 \times 1 \times 128$	dropout 20%	0
$1 \times 1 \times 128$	fc + softmax	258
$1 \times 1 \times 2$	saída	Total: 10.579.394

Fonte: Autoria própria (2024).

operação *flatten*, que simplesmente converte todos os valores do tensor em uma representação unidimensional. Isso força uma correspondência entre os *feature maps* e as classes de saída. Para compensar a redução de parâmetros, aumentou-se a profundidade da rede, reduzindo progressivamente a dimensão da imagem. Nas últimas camadas convolucionais, utilizou-se um módulo Inception tal como o apresentado anteriormente na Figura 6a.

Tabela 7 – Arquitetura da rede totalmente convolucional. A redução de dimensão é feita por convoluções com passo (*stride*) 2, e usa-se GAP após a última convolução. Nas camadas convolucionais, também se utiliza *batch normalization*.

Entrada	Operação	Parâmetros treináveis
$128 \times 320 \times 3$	conv 3x3 + relu + bn	480
$128 \times 320 \times 16$	conv 3x3 stride 2x2 + relu + bn	2.352
$64 \times 160 \times 16$	conv 3x3 + relu + bn	4.704
$64 \times 160 \times 32$	conv 3x3 stride 2x2 + relu + bn	9.312
$32 \times 80 \times 32$	conv 3x3 + relu + bn	18.624
$32 \times 80 \times 64$	conv 3x3 stride 2x2 + relu + bn	37.056
$16 \times 40 \times 64$	conv 3x3 + relu + bn	74.112
$16 \times 40 \times 128$	conv 3x3 stride 2x2 + relu + bn	147.840
$8 \times 20 \times 128$	conv 3x3 + relu + bn	147.840
$8 \times 20 \times 128$	conv 3x3 stride 2x2 + relu + bn	147.840
$4 \times 10 \times 128$	inception-v2a	151.440
$4 \times 10 \times 256$	gap	0
$1 \times 1 \times 256$	fc + softmax	514
$1 \times 1 \times 2$	saída	Total: 742.114

Fonte: Autoria própria (2024).

Também foram feitos experimentos com arquiteturas tradicionais: MobileNetV2, EfficientNetB0 e DenseNet-121 (ver seção 2.2), que, devido às suas otimizações, ainda são viáveis em algumas plataformas embarcadas. Normalmente há disponibilidade de pesos pré-treinados em grandes datasets para essas arquiteturas, que podem ser utilizados como pesos iniciais para um treinamento em outro domínio, conforme descrito na subseção 2.1.6.

5 EXPERIMENTOS E RESULTADOS

Buscando verificar a viabilidade do uso dos dados sintéticos para o aprendizado de modelos de classificação do estado de chaves seccionadoras, foram realizados experimentos com as seis diferentes arquiteturas de redes neurais convolucionais elencadas na seção 4.9, realizando múltiplas rodadas de treinamento com diferentes números de amostras. Também se avaliou a inclusão de uma pequena porção de dados reais para promover o reconhecimento de características comuns aos domínios sintético e real, reduzindo o *domain gap* entre eles.

A implementação da solução proposta foi realizada tendo em vista a implantação no computador embarcado NVIDIA Jetson Nano[1], especializado para a execução de inferência com algoritmos de inteligência artificial utilizando computação paralela em GPU. Seus principais recursos são a arquitetura de GPU com 128 núcleos, 4 GB de memória, conectividade de rede Gigabit Ethernet e tamanho físico de 100mm x 80mm. O produto também conta com suporte de *software* para bibliotecas como TensorFlow[2] e OpenCV, permitindo que as soluções sejam desenvolvidas em computadores tradicionais e instaladas com nenhuma ou pouca adaptação no *hardware* final.

5.1 Formação de conjuntos de imagens sintéticas

Para o treinamento dos modelos de visão computacional, realizou-se a geração de conjuntos de imagens sintéticas de chaves seccionadoras utilizando o *framework* de renderização (seção 4.7) e extração de amostras (seção 4.8). Foram geradas imagens de todos os *presets* virtuais, utilizando os seguintes parâmetros (determinados empiricamente):

- Horário entre 6h15min e 17h45min, amostrados aleatoriamente de uma distribuição uniforme;
- Inclinação da iluminação direcional entre $0°$ e $25°$ (aleatória, uniforme);
- Intensidade da luz entre 20% e 100% (aleatória, uniforme);
- Perturbação de posição da câmera, amostrada de uma distribuição normal com $\mu = 0\,\text{cm}$ e $\sigma = 1\,\text{cm}$ para cada um dos três eixos cartesianos;
- Perturbação de rotação da câmera, amostrada de uma distribuição normal com $\mu = 0°$ e $\sigma = 0{,}029°$ para os ângulos de Euler referentes aos eixos x e z, e $\mu = 0°$ e $\sigma = 0{,}057°$ para o eixo y;
- Grau de abertura das chaves do enquadramento entre 0,00 e 0,02 para chaves fechadas e entre 0,98 e 1,00 para chaves abertas (aleatório, uniforme).

[1] https://www.nvidia.com/pt-br/autonomous-machines/embedded-systems/jetson-nano/
[2] https://www.tensorflow.org/

- Brita habilitada em 10% das renderizações (aleatória, uniforme).

A partir das renderizações, fez-se a geração de amostras juntamente à adição de um plano de fundo aleatoriamente selecionado e manipulado (recorte, espelhamento, rotação, borramento, brilho e contraste). A Tabela 8 consolida a quantidade total de amostras sintéticas produzidas para cada tipo de chave e estado. As categorias nomeadas “abertura central 1” e “abertura central 2” se referem a duas variantes de chaves do tipo abertura central existentes na Subestação Campo Comprido e anteriormente ilustradas na Figura 21. Julgou-se necessário fazer essa distinção, pois as diferenças de aspecto entre as duas se concentram na parte móvel da chave e nas proximidades do contato, ou seja, na região da imagem mais importante para o reconhecimento. Quanto aos demais tipos de chave, não se considerou necessário separar suas variantes, pois as diferenças de aspecto se resumem a detalhes nas partes imóveis da chave.

Tabela 8 – Número de amostras sintéticas produzidas para o treinamento das redes convolucionais, para cada tipo de chave e estado. Buscou-se gerar amostras em quantidade excedente para que os treinamentos pudessem ser realizados com subpartições de tamanho igual para todos os tipos de chave.

Tipo de chave	Estado	Amostras
Abertura central 1	aberta	110.700
Abertura central 1	fechada	110.700
Abertura central 2	aberta	102.000
Abertura central 2	fechada	102.000
Dupla abertura	aberta	117.000
Dupla abertura	fechada	117.000
Semi-pantográfica vertical	aberta	105.000
Semi-pantográfica vertical	fechada	105.000
Semi-pantográfica horizontal	aberta	139.500
Semi-pantográfica horizontal	fechada	139.500
Total		1.148.400

Fonte: Autoria própria (2024).

5.2 Treinamento e teste dos modelos de aprendizado de máquina

A implementação das arquiteturas propostas na seção 4.9 (simples, simples expandida e totalmente convolucional) foi realizada em linguagem Python, com o auxílio das bibliotecas TensorFlow e Keras[3]. Para essas arquiteturas, utilizou-se inicialização aleatória de pesos.

Quanto às arquiteturas MobileNetV2, EfficientNetB0 e DenseNet-121, utilizaram-se as implementações disponibilizadas na biblioteca Keras, inicializadas com pesos pré-treinados no *dataset* ImageNet. No treinamento desses modelos para o reconhecimento do estado de chaves, congelaram-se os pesos das 30% primeiras camadas, preservando, assim, parte dos filtros

[3] https://keras.io/

pré-treinados no início da rede. Em experimentos iniciais, foram testadas outras porcentagens, incluindo o treinamento completo da rede, mas o desempenho observado foi prejudicado.

Modelos distintos foram treinados para cada tipo de chave: abertura central, dupla abertura, semi-pantográfica vertical e semi-pantográfica horizontal, utilizando as amostras sintéticas listadas na Tabela 8. Todos os treinamentos foram realizados com 24 épocas, usando a função de otimização Adam com taxa de aprendizado 0,0001 e a função de perda *categorical cross-entropy*. Para validação, fez-se uma separação aleatória de 20% dos dados de treinamento. Durante o treinamento, o salvamento dos pesos é feito por meio de *checkpoint*, ou seja, ao final de cada época, avalia-se a acurácia do modelo no conjunto de validação e, se houve melhora, os pesos são salvos. Assim, o modelo salvo durante um treinamento é sempre aquele que apresentar a melhor acurácia medida para o conjunto de validação ao final de uma de suas épocas.

Para o teste dos modelos, usaram-se as amostras obtidas a partir dos conjuntos de capturas das câmeras físicas nas subestações, descritos anteriormente na Tabela 3. A Tabela 9 quantifica as amostras de cada conjunto de teste: “Normal 1”, “Normal 2” e “Manobras”, esse último contendo imagens das chaves em seu estado oposto ao usual. Houve um intervalo de aproximadamente 3 meses entre as coletas das imagens dos conjuntos Normal 1 e Normal 2. Nos testes, os três conjuntos são considerados de maneira independente — no caso das imagens com as chaves no seu estado usual, para observar se existem variações devido a mudanças ocorridas com o passar do tempo; e no caso das imagens de manobras, porque o seu número é bastante reduzido, ou seja, representam um grupo minoritário. Como a avaliação é feita para cada conjunto em separado, a métrica de acurácia pode ser utilizada, já que não há risco de penalizar os resultados sobre as classes minoritárias.

5.3 Influência da quantidade de dados sintéticos

Para validar o uso dos dados sintéticos no treinamento de modelos para classificar dados reais e, adicionalmente, identificar a variação de desempenho em função do tamanho do conjunto de treinamento, foram realizados treinamentos com diferentes quantidades de amostras sintéticas. Este experimento foi realizado com partições de tamanhos progressivamente maiores dos dados enumerados na Tabela 8, com cada partição tendo desde 100 até 150.000 amostras, escolhidas aleatoriamente para cada instância de treinamento. Como cada treinamento possui fatores não determinísticos (tais como o particionamento do conjunto de treinamento e a inicialização de pesos), foram realizadas três repetições para cada configuração única de treinamento, sendo considerada como resultado a acurácia média entre as três execuções.

Em um primeiro momento, os experimentos foram feitos com imagens de chaves do tipo “abertura central 1”. A Figura 48 apresenta os resultados de acurácia média sobre os conjuntos Normal 1 e Normal 2 para as arquiteturas simples e simples expandida (*vide* seção 4.9) em um gráfico de linhas. Para aprimorar a visualização, utilizou-se escala logarítmica no eixo de amos-

Tabela 9 – Número de amostras reais utilizadas nos testes das redes convolucionais para cada tipo de chave e estado.

		Dataset		
Tipo de chave	Estado	Normal 1	Normal 2	Manobras
Abertura central 1	aberta	2.310	400	85
Abertura central 1	fechada	16.584	4.626	6
Abertura central 2	aberta	990	315	12
Abertura central 2	fechada	2.321	761	6
Dupla abertura	aberta	6.758	2.752	59
Dupla abertura	fechada	11.425	2.361	33
Semi-pantográfica vertical	aberta	2.046	423	0
Semi-pantográfica vertical	fechada	0	0	12
Subtotal (CCO)		42.434	11.638	213
Semi-pantográfica horizontal	aberta	0	0	94
Semi-pantográfica horizontal	fechada	4.148	3.443	95
Subtotal (BTA)		4.148	3.443	189
Total		46.582	15.081	402

Fonte: Autoria própria (2024).

tras, já que os ganhos de acurácia são progressivamente menores à medida que se aumenta a quantidade de dados.

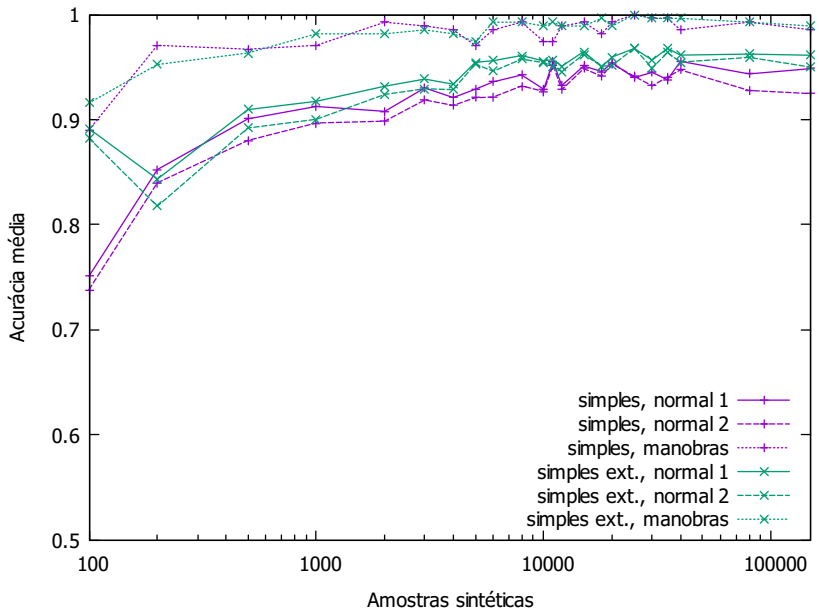


Figura 48 – Desempenho médio dos modelos treinados para chaves do tipo “abertura central 1” em função da quantidade de amostras sintéticas.

Fonte: Autoria própria (2024).

Observa-se que, para as seccionadoras de abertura central, pequenas quantidades de amostras sintéticas de treinamento (cerca de 1.000) já são capazes de produzir modelos com acurácia próxima a 90%. O ganho de acurácia ao aumentar a quantidade de dados desacelera

a partir de 20.000 amostras para ambas as arquiteturas, atingindo cerca de 96%; mesmo assim, não se observa prejuízo no desempenho ao continuar a aumentar o número de amostras. Comparando as curvas produzidas pelas duas arquiteturas sobre cada conjunto de teste, nota-se que os resultados para o conjunto Normal 1 e Normal 2 foram semelhantes (com leve vantagem para o Normal 1), e que o desempenho em manobras foi significativamente superior a ambos ao longo de todo o experimento. Entre as duas arquiteturas testadas, há uma vantagem para a simples expandida.

O experimento também foi realizado para as imagens de chaves de dupla abertura. A acurácia média dos modelos treinados é ilustrada na Figura 49.

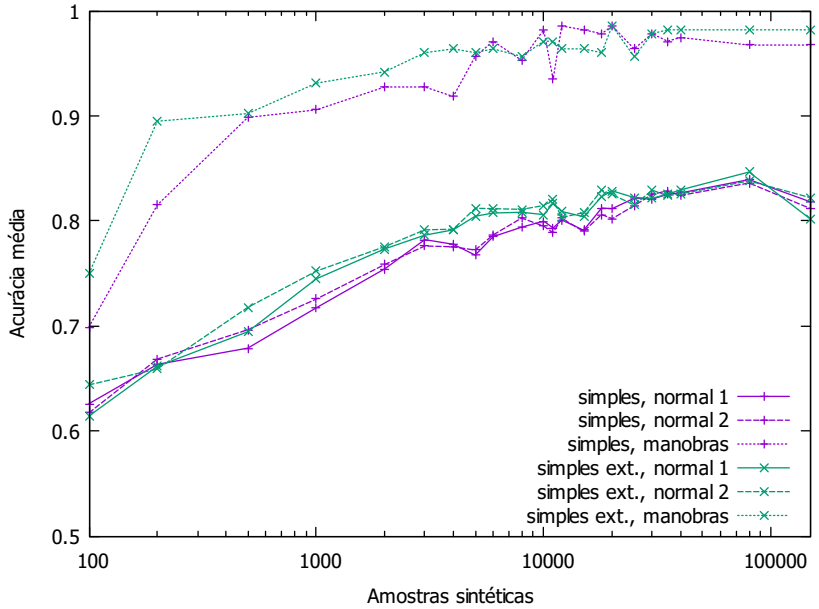


Figura 49 – Desempenho médio dos modelos treinados para chaves do tipo “dupla abertura” em função da quantidade de amostras sintéticas.

Fonte: Autoria própria (2024).

Em comparação ao experimento com chaves de abertura central, todas as curvas de desempenho estão mais baixas para as seccionadoras de dupla abertura, não conseguindo ultrapassar 85% de acerto nos conjuntos Normal 1 e Normal 2. Isso evidencia que esta instância do problema é mais difícil de ser resolvida. Com relação à variação da quantidade de dados, ainda aparenta existir uma pequena tendência crescente na acurácia após 40.000 amostras. Pode-se afirmar que, para este tipo de chave, são necessárias mais imagens sintéticas de treinamento do que para chaves do tipo “abertura central 1”.

As duas categorias de chave apresentadas nesta seção foram escolhidas por terem a maior disponibilidade de dados de teste, a fim de promover uma análise mais precisa. Para os demais tipos de chaves, o comportamento geral foi similar a essas observações.

Em suma, os resultados dos experimentos mostram que os modelos treinados com dados sintéticos conseguem, com limitações, ser aplicados no domínio real e alcançar taxas de acerto elevadas. Entretanto, o comportamento não é homogêneo entre os diferentes tipos de

chaves: no caso da abertura central, pequenas quantidades de dados de treinamento (na ordem de 1.000) já se mostraram suficientes para superar a acurácia de 90%, enquanto que para a dupla abertura, esse patamar não foi alcançado mesmo com 150 mil amostras. Além disso, observou-se que há um teto de desempenho (cujo ponto de início também difere por chave), a partir do qual o desempenho se mantém estável, sem crescimento aparente ao continuar a adicionar amostras de treinamento.

5.4 Comparação do desempenho dos modelos treinados apenas com dados sintéticos

Nos experimentos anteriores, não se observou prejuízo em aumentar a quantidade de dados de treinamento até o maior valor utilizado (150.000 amostras). Além disso, a quantidade de dados necessária para a estabilização do desempenho difere para cada tipo de chave. Assim, para comparar os resultados entre chaves de tipos diferentes, consideraram-se apenas os modelos treinados com 150.000 amostras sintéticas.

5.4.1 Comparação por acurácia média

A Figura 50 apresenta a acurácia média entre as três repetições de cada configuração de treinamento para as diferentes chaves e redes neurais convolucionais. Os mesmos resultados são, também, apresentados em forma tabular no Apêndice A (Tabela 12).

Todos os processos de treinamento atingiram acurácia de validação maior que 98%, alguns inclusive atingindo 100%, confirmando que os modelos conseguiram identificar as características relevantes para a classificação no domínio sintético. Contudo, ao comparar a acurácia de cada modelo no conjunto de validação com os resultados obtidos nos três conjuntos de teste, observa-se, de forma geral, uma redução, o que evidencia a existência de um *domain gap* entre os dados sintéticos e os dados reais, como já esperado.

Para cada arquitetura, observa-se que, em geral, as taxas de acerto para os *datasets* Normal 1 e Normal 2 foram bastante próximas, por mais que os dados tenham sido coletados em épocas diferentes. Esse comportamento é coerente com o fato de que não foi apresentada nenhuma imagem real para as redes neurais durante o treinamento. Além disso, para todos os tipos de chave exceto a semi-pantográfica vertical (SPV), os modelos obtiveram melhor desempenho no reconhecimento do estado no conjunto Manobras do que nos conjuntos Normal 1 e 2. Isso mostra que as variações de abertura da chave presentes nas amostras sintéticas contribuíram para que o modelo identificasse as manobras de chaves reais. A diferença de desempenho entre as imagens no estado normal e as imagens no estado manobrado pode ser atribuída ao fato de que a realização de manobras, devido às limitações da própria atividade, ocorreu apenas no período diurno e em condições de boa visibilidade.

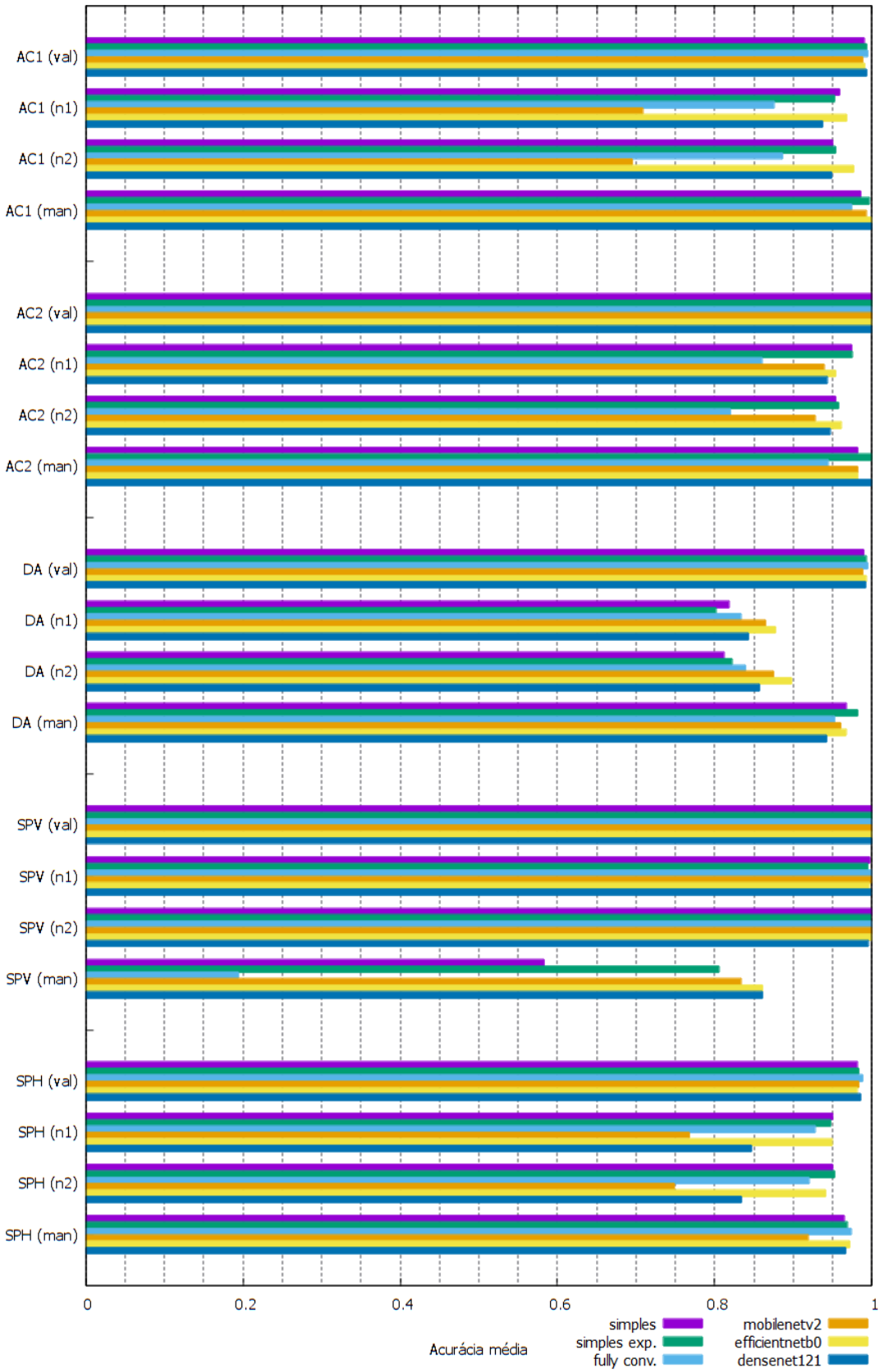


Figura 50 – Desempenho médio dos modelos treinados com 150.000 amostras sintéticas. O desempenho sobre o conjunto de validação, composto por imagens sintéticas, está incluso como referência comparativa frente aos conjuntos de amostras reais.

Fonte: Autoria própria (2024).

No caso específico das chaves SPV, todos os modelos foram capazes de reconhecer o estado das chaves nos conjuntos Normal 1 e 2 com alta acurácia (mais de 99%), mas o teste dos modelos sobre o conjunto de manobras não foi positivo: a melhor acurácia média alcançada foi cerca de 86%, e duas arquiteturas sequer ultrapassaram 60%. Como há poucas chaves desse tipo na subestação, o conjunto de teste contempla apenas 15 pontos de vista diferentes, e essa limitação é ainda mais severa no conjunto de manobras, composto por somente 12 amostras, dificultando a conclusão acerca dos resultados obtidos. Uma possível explicação é que, para essa categoria de chave, todas as chaves da subestação estão normalmente no estado aberto; assim, o teste do reconhecimento do estado fechado apenas pôde ser realizado sobre o conjunto de manobras. Isso pode indicar que as amostras sintéticas de chaves fechadas não são suficientemente parecidas com as capturas reais das chaves manobradas.

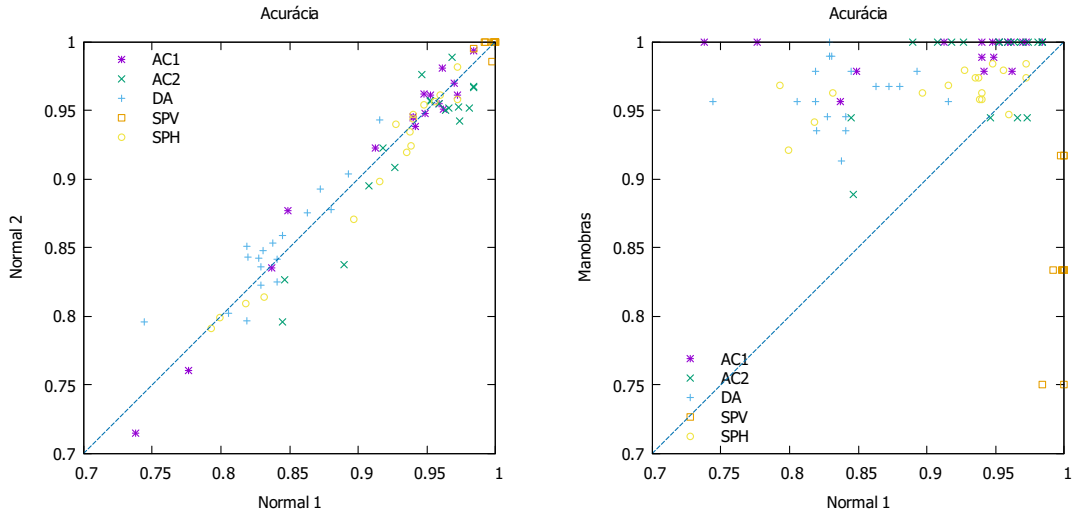
Para as demais chaves, os maiores resultados obtidos no conjunto Normal 1 foram 97,49% (abertura central 2, arquitetura simples expandida), 96,81% (abertura central 1, arquitetura EfficientNetB0), 95,03% (semi-pantográfica horizontal, arquitetura simples) e 87,77% (dupla abertura, arquitetura EfficientNetB0). Em muitos casos, as arquiteturas elaboradas como *baseline* nesta pesquisa (simples e simples expandida) superaram os resultados das três redes neurais bem-conhecidas (MobileNetV2, EfficientNetB0 e DenseNet-121), sugerindo que as redes maiores, por terem mais graus de liberdade, são mais sujeitas a modelar características que existem apenas nas imagens sintéticas. Nas chaves de dupla abertura (DA), embora o resultado em manobras tenha alcançado a ordem de 95%, nenhuma das arquiteturas conseguiu ultrapassar 90% de acurácia média sobre as amostras de chave na situação normal. Isso sugere que essa categoria de chaves é mais desafiadora para a transferência de aprendizado, seja por conta da geometria dos equipamentos ou de pontos de vista menos favoráveis.

Nas chaves “abertura central 1” (AC1), “abertura central 2” (AC2) e “semi-pantográfica horizontal” (SPH), observa-se que algumas arquiteturas, especialmente a totalmente conectada e a MobileNetV2, apresentaram resultados significativamente inferiores às outras arquiteturas nos conjuntos Normal 1 e 2, embora o desempenho sobre o conjunto de validação tenha sido bom. Este não é um efeito recorrente — fazendo treinamentos adicionais com configurações idênticas, foi possível obter resultados equiparáveis aos das demais arquiteturas; entretanto, para manter a comparação justa, o gráfico ilustra apenas os três primeiros treinamentos. Mesmo assim, isso demonstra que um modelo treinado apenas em dados sintéticos pode se degenerar e não funcionar no domínio real, possivelmente devido a uma inicialização de pesos desfavorável no processo de treinamento.

5.4.2 Comparação entre rodadas de treinamento individuais

Além do resultado médio entre execuções com a mesma configuração, também se avaliaram os resultados para cada tipo de chave por meio de gráficos de dispersão, nos quais cada ponto representa o resultado de uma execução única de treinamento (sem média). A Figura 51

apresenta dois gráficos cujos eixos representam o desempenho sobre um dos três conjuntos de teste: Normal 1, Normal 2 e Manobras. A linha tracejada é a diagonal do gráfico, servindo apenas referência para comparar se o modelo teve melhor desempenho em um dos dois conjuntos — em um caso ideal, sem vieses, os pontos devem se concentrar bem próximos a esta diagonal. Além disso, os resultados são melhores quanto mais próximos do canto superior direito estiverem. Para cada tipo de chave, há 18 pontos, referentes às 3 repetições do treinamento com 150.000 amostras para cada uma das 6 arquiteturas (sem distinção).



(a) Normal 1 x Normal 2 **(b) Normal 1 x Manobras**

Figura 51 – Desempenho de cada modelo treinado com 150.000 amostras sintéticas, para cada tipo de chave. Em cada gráfico, compara-se o desempenho sobre o conjunto Normal 1 com um dos outros. A escala dos eixos inicia em 70%.

Fonte: Autoria própria (2024).

Os pontos na Figura 51a estão distribuídos de forma equilibrada em torno da diagonal do gráfico, indicando não haver desvios de desempenho dos modelos sobre os conjuntos Normal 1 e Normal 2. Já na Figura 51b, os pontos em sua maioria sobem para a região acima da diagonal, mostrando que o desempenho no reconhecimento de manobras é superior ao reconhecimento do estado na situação normal. Os pontos referentes às chaves SPV, entretanto, mostram o efeito contrário: taxas de acerto muito elevadas na situação normal e quedas bruscas de desempenho em manobras (inclusive com alguns pontos abaixo da escala do gráfico). Como levantado anteriormente, todas as chaves SPV da subestação estudada estão normalmente no estado aberto, e a baixa taxa de acerto no estado fechado pode ser atribuída a diferenças de aspecto entre as imagens sintéticas de chaves fechadas e as imagens reais capturadas em manobras.

Analisando novamente o gráfico da Figura 51a, os pontos referentes às chaves AC1 e AC2 se concentram próximos a 95% para ambos os conjuntos, enquanto que para as chaves SPH isso ocorre um pouco abaixo, próximo a 93%. Já para as chaves DA, a concentração se localiza próxima a 84%, novamente mostrando a maior dificuldade dessa instância do problema. Também é possível observar que há bastante espalhamento nos pontos de todas as chaves (ex-

ceto SPV), o que mostra que um bom desempenho em dados sintéticos (validação) nem sempre se converte em um bom desempenho no domínio real. De modo geral, os resultados dos modelos nos testes com dados reais sugerem que a abordagem de transferência de aprendizado a partir de um domínio totalmente sintético é promissora, embora ainda exista oportunidade de melhoramento.

5.5 Influência da adição de dados reais no treinamento

Visando combater o *domain gap*, especialmente nos casos das chaves para as quais os modelos treinados apenas sobre dados sintéticos não alcançaram bons resultados, avaliou-se a inclusão de dados reais em pequena quantidade no treinamento. Para isso, formou-se um novo *dataset* a partir de imagens diferentes daquelas utilizadas para testar os modelos (descritas anteriormente na Tabela 9). No caso das chaves da SE CCO, as imagens foram capturadas na data de 19/02/2022, aproximadamente 21 dias antes do conjunto Normal 1. Já as imagens da SE BTA foram capturadas na data de 07/03/2022, a qual está dentro do intervalo de datas do *dataset* Normal 1, mas em uma faixa de horário que não foi incluída no conjunto de teste. De forma geral, as imagens reais adicionadas no treinamento são de períodos próximos ao conjunto de testes Normal 1, e precedem o conjunto Normal 2 por cerca de três meses.

A composição do novo *dataset* está elencada na Tabela 10. Como se pode observar, o número total de amostras reais, 6.116, é bastante reduzido se comparado à quantidade de amostras sintéticas geradas, 1.148.400. Além disso, neste conjunto, todas as imagens de uma mesma chave a mostram no mesmo estado; por isso, durante o treinamento, a variações de estado apresentadas aos modelos são provenientes apenas das imagens sintéticas. Outro ponto importante é que não há amostras de chaves semi-pantográficas verticais fechadas, nem de semi-pantográficas horizontais abertas. Isto ocorre porque, no conjunto de chaves monitorado, todas as chaves destes tipos estão normalmente no mesmo estado.

O método escolhido para incluir essas imagens no treinamento foi o de treinamento conjunto: as imagens são concatenadas aos dados sintéticos de treinamento e apresentadas simultaneamente à rede neural convolucional. A partição de validação é composta apenas por dados sintéticos. Os demais parâmetros do algoritmo de treinamento foram mantidos idênticos aos utilizados nos experimentos anteriores, a fim de permitir uma melhor comparação dos resultados. Assim como nos experimentos da seção 5.3, considerou-se o resultado a partir da acurácia medida para três rodadas de treinamento.

5.5.1 Comparação por acurácia média

Para observar a influência dos dados reais diante de diferentes quantidades de dados sintéticos, foi realizado inicialmente um experimento que contemplou múltiplas rodadas de trei-

Tabela 10 – Conjunto de dados adicionais para treinamento contendo amostras extraídas de imagens reais.

Tipo de chave	Estado	Amostras
Abertura central 1	aberta	593
Abertura central 1	fechada	1.965
Abertura central 2	aberta	124
Abertura central 2	fechada	289
Dupla abertura	aberta	843
Dupla abertura	fechada	1.501
Semi-pantográfica vertical	aberta	312
Semi-pantográfica vertical	fechada	0
Semi-pantográfica horizontal	aberta	0
Semi-pantográfica horizontal	fechada	489
Total		6.116

Fonte: Autoria própria (2024).

namento com quantidades crescentes de dados sintéticos (assim como nos testes relatados na seção 5.3), mas considerando somente as chaves da classe “dupla abertura”. Este tipo de seccionadora foi selecionado para este teste por ter sido o caso que se mostrou mais desafiador nos testes anteriores. Em todos os treinamentos, a quantidade de dados reais foi mantida fixa (conforme a Tabela 10); isso significa que as execuções com menos de 2.344 amostras sintéticas foram feitas, na verdade, com mais dados reais que sintéticos. A Figura 52 apresenta os resultados do experimento.

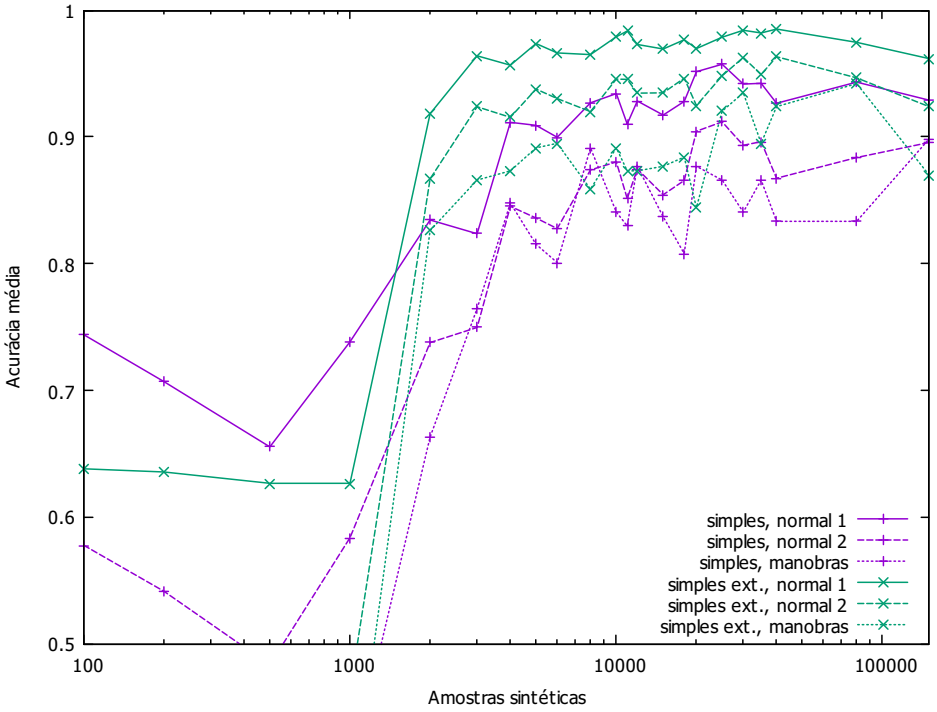
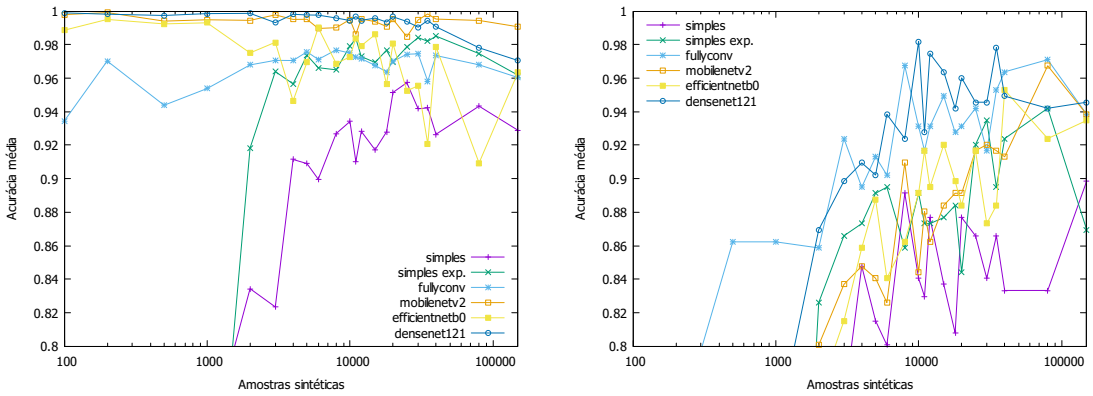


Figura 52 – Desempenho médio dos modelos treinados com dados reais e sintéticos para chaves do tipo “dupla abertura”, em função da quantidade de amostras sintéticas.

Fonte: Autoria própria (2024).

Em comparação com o treinamento sem dados reais (Figura 49), nota-se que houve uma melhora substancial no desempenho nas imagens referentes à situação normal das chaves. A arquitetura simples expandida, por exemplo, conseguiu ultrapassar 95% de acerto no conjunto Normal 1 com apenas 2.000 amostras sintéticas — sem as amostras reais, mesmo com 150.000 amostras sintéticas, a taxa de acerto nunca passou de 85%. Todavia, a taxa de acerto em manobras foi afetada negativamente, estagnando pouco abaixo de 90%. Outro efeito observado é que a melhora no desempenho no conjunto Normal 2, apesar de também ter sido positiva, ocorreu em uma proporção menor que no conjunto Normal 1, o que pode ser explicado pela maior proximidade das datas de captura das imagens reais de treinamento em relação às fotografias do Normal 1. Essas observações mostram que a inclusão de imagens reais no treinamento provocou algum grau de tendência no resultado do modelo.

Outro comportamento observado é que, para quantidades inferiores a 2.000 imagens sintéticas, adicionar as 2.344 amostras reais de treinamento piorou significativamente o desempenho nos três conjuntos de teste (nas arquiteturas simples e simples expandida). Já para as demais arquiteturas, esse efeito foi observado apenas no conjunto de manobras, como mostrado na Figura 53. Nas manobras, há, ainda, um comportamento bastante errático ao longo de todas as curvas de desempenho, indicando novamente que a inclusão de imagens reais na situação normal não favoreceu o reconhecimento de manobras. Uma possível causa para tal comportamento é o fato de que, nas amostras reais apresentadas no treinamento, cada chave aparece sempre em um mesmo estado, que é o mesmo observado nos conjuntos Normal 1 e Normal 2. Desta forma, é possível que as redes tenham aprendido padrões que só existem nas fotografias, mas não nas amostras sintéticas, associando assim tais padrões ao estado usual de cada chave.



(a) Normal 1 **(b) Manobras**

Figura 53 – Resultado das demais arquiteturas no experimento com treinamento conjunto entre dados reais e sintéticos para chaves de dupla abertura.
Fonte: Autoria própria (2024).

O experimento também foi realizado para os demais tipos de chaves, mas com o número de amostras sintéticas fixo em 150.000. A Figura 54 ilustra os resultados médios obtidos neste experimento, também apresentados em forma tabular no Apêndice A (Tabela 13). Ao compará-

la com o gráfico do experimento anterior (Figura 50), é possível perceber que houve melhora significativa no desempenho sobre os conjuntos Normal 1 e 2.

Também se pode verificar que a acurácia de validação permaneceu praticamente inalterada entre os dois experimentos. Assim, o aumento da acurácia sobre os conjuntos Normal 1 e 2 representa um estreitamento do *domain gap*. Além disso, a grande diferença que havia sido observada entre os resultados de redes neurais distintas para a mesma chave (especialmente a totalmente conectada a MobileNetV2) deixou de ocorrer, mostrando que a inclusão de dados reais no treinamento promoveu a extração de características úteis para o reconhecimento em ambos os domínios.

Outra constatação é que, após a inclusão de dados reais, os melhores resultados passaram a ser produzidos pelas arquiteturas bem-conhecidas: as redes neurais convolucionais MobileNetV2, EfficientNetB0 e DenseNet-121. Para cada tipo de chave exceto semi-pantográfica vertical, as maiores taxas de acerto sobre o conjunto Normal 1 foram 99,95% (abertura central 2, DenseNet-121), 99,91% (abertura central 1, DenseNet-121), 99,06% (dupla abertura, MobileNetV2) e 97,81% (semi-pantográfica horizontal, EfficientNetB0). Os resultados superam aqueles relatados nos trabalhos de Wang (2018) (91,3%) e Quan *et al.* (2022) (91,12%). Embora não diretamente comparável, também se equiparam aos de Le, Pham e Phung (2022) (mAP de 0,9974), mas com testes significativamente mais extensos em quantidade e diversidade de amostras.

5.5.2 Comparação entre rodadas de treinamento individuais

A Figura 55 apresenta o resultado de cada instância de treinamento em gráficos de dispersão, assim como ilustrado nos experimentos anteriores (Figura 51). Comparativamente, a Figura 55a mostra que houve uma melhora significativa no desempenho de todos os modelos nos conjuntos de amostras normais ao adicionar imagens reais no treinamento: apenas três pontos (todos referentes às chaves de dupla abertura) entre os 90 treinamentos obtiveram taxa de acerto menor que 90% em ao menos um dos *datasets*. Adicionalmente, o espalhamento dos pontos foi reduzido em todas as chaves, ou seja, os modelos passaram a ser mais consistentes em obter bons resultados no reconhecimento de imagens reais na situação usual das chaves. Neste experimento, as concentrações de pontos se localizam aproximadamente em 99% para as chaves de abertura central 1 (AC1), 98% para as chaves de abertura central 2 (AC2) e 96% para as chaves semi-pantográficas horizontais (SPH). Nos treinamentos com chaves de dupla abertura (DA), há um espalhamento maior em contraste com as outras chaves, mas ainda melhor que nos experimentos anteriores (apenas com dados sintéticos), com centroide aproximadamente em 95% em ambos os eixos.

Ainda na Figura 55a, nota-se que a maioria dos pontos está abaixo da diagonal, o que indica que os modelos passaram a ter melhor resultado sobre o conjunto Normal 1 do que sobre o Normal 2 após adicionar as amostras reais ao treinamento, especialmente para as

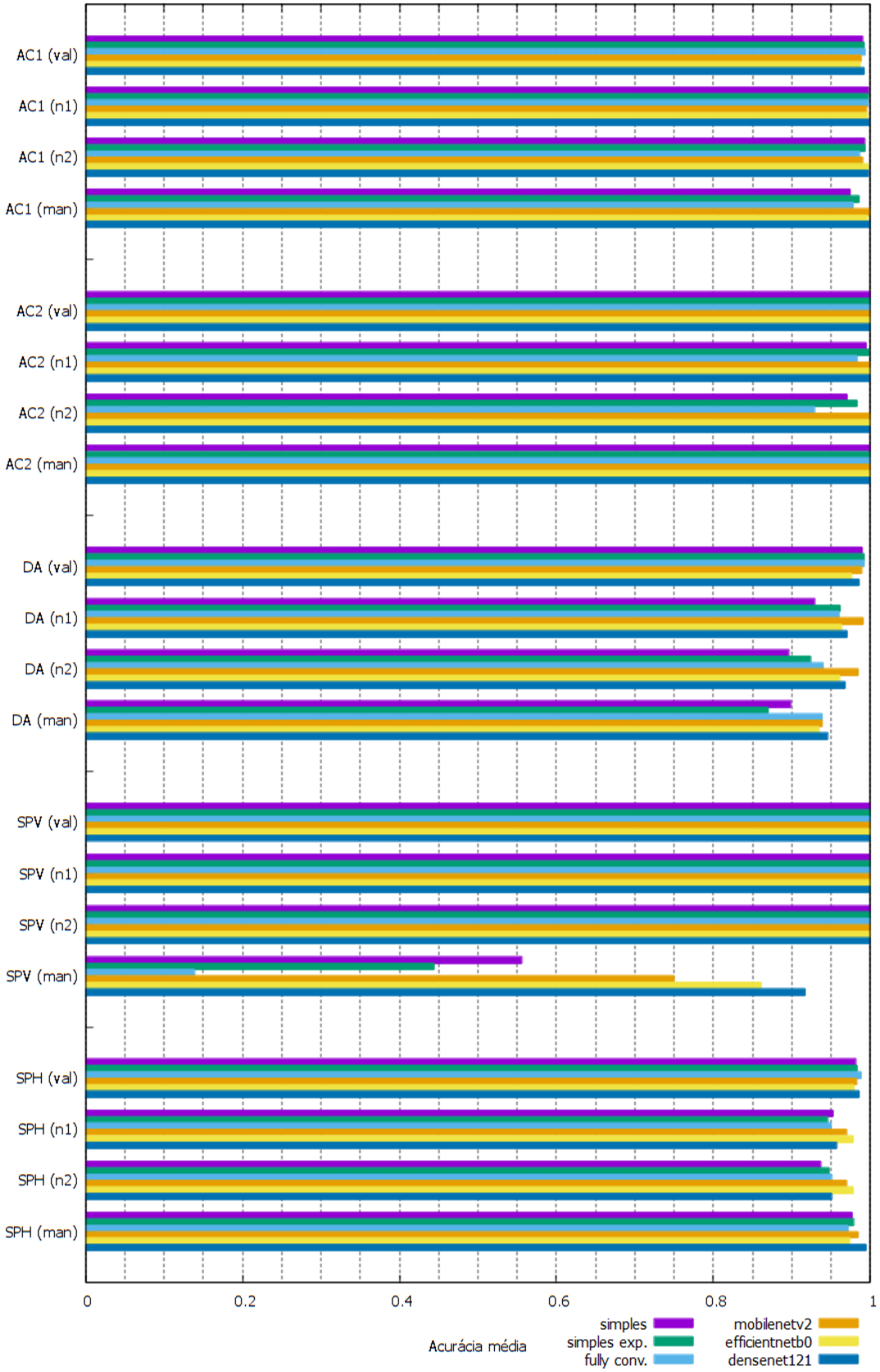
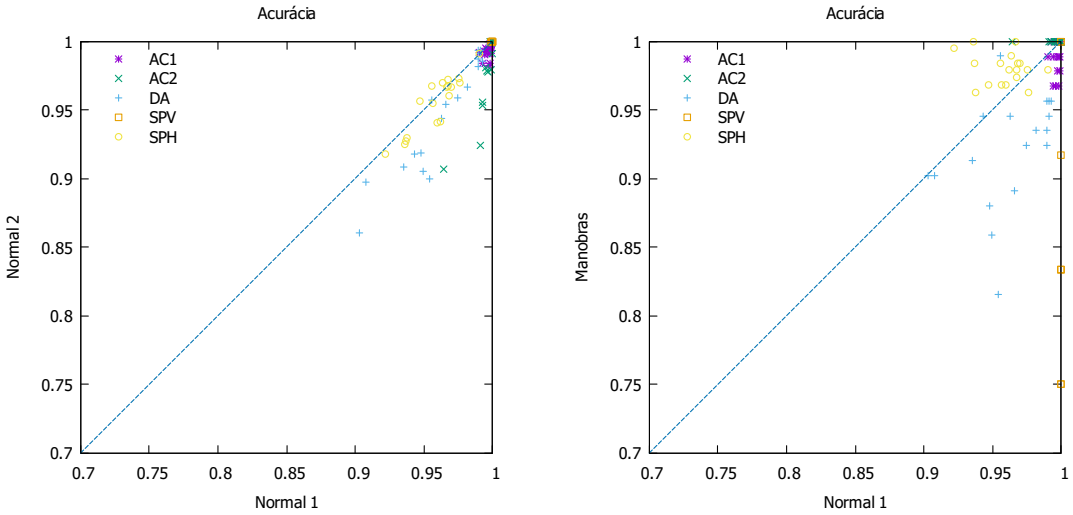


Figura 54 – Desempenho médio dos modelos treinados com 150.000 amostras sintéticas juntamente com dados reais. O desempenho sobre o conjunto de validação, composto por imagens sintéticas, está incluso como referência comparativa frente aos conjuntos de amostras reais.

Fonte: Autoria própria (2024).



(a) Normal 1 x Normal 2 **(b) Normal 1 x Manobras**

Figura 55 – Desempenho de cada modelo treinado com 150.000 amostras sintéticas, juntamente com as amostras reais listadas na Tabela 10, para cada tipo de chave. Em cada gráfico, compara-se o desempenho sobre o conjunto Normal 1 com um dos outros. A escala dos eixos inicia em 70%.

Fonte: Autoria própria (2024).

chaves AC2 e DA. Esse comportamento possivelmente se deve ao fato de que as imagens reais de treinamento foram obtidas em períodos próximos aos das imagens do Normal 1, tendo características mais semelhantes a esse *dataset* do que ao outro. Na Figura 56, mostram-se dois exemplos de mudanças no ambiente das subestações ocorridas entre os dois conjuntos de teste — no Normal 2, as imagens têm, em geral, iluminação menos intensa, direções de sombra diferentes no mesmo horário, vegetação menos verdejante, além de diferenças na situação dos próprios equipamentos da subestação.

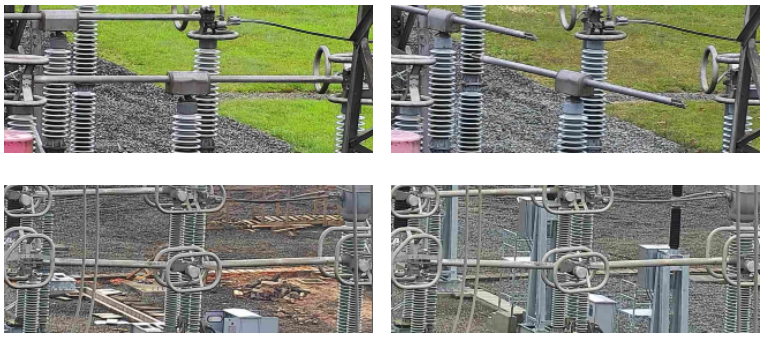


Figura 56 – Exemplos de diferenças de aspecto entre imagens dos conjuntos Normal 1 (à esquerda) e Normal 2 (à direita). As imagens reais de treinamento são mais parecidas com as do Normal 1, o que pode explicar o pequeno desvio de resultado entre os conjuntos Normal 1 e 2.

Fonte: Autoria própria (2024).

Comparando, agora, a Figura 55b com sua equivalente nos experimentos anteriores (Figura 51b), observa-se que, em geral, os pontos apenas se movimentaram para a direita, ou seja, não houve melhora no reconhecimento de manobras. Isso se deve ao fato de que

não há chaves manobradas no *dataset* real de treinamento: estas imagens sempre mostram cada chave em seu estado usual, assim como nos *datasets* de teste Normal 1 e 2. Ademais, no caso específico das seccionadoras do tipo DA, é possível notar uma redução na acurácia sobre o conjunto de manobras após adicionar as amostras reais no treinamento, o que pode estar relacionado à maior dificuldade inerente a esse tipo de chave. Isso não se observa nas categorias de chaves SPH e AC2, nas quais o desempenho em manobras permaneceu melhor que o desempenho sobre as imagens na situação normal.

A anomalia observada no experimento anterior para as seccionadoras do tipo semi-pantográfica vertical (SPV) não foi resolvida com a adição de amostras reais no treinamento. Todos os treinamentos alcançaram taxas de acerto superiores a 99,8% em ambos os conjuntos de amostras normais, ainda superiores aos valores obtidos no treinamento somente com dados sintéticos. Por outro lado, o desempenho em manobras para este tipo de chave permaneceu instável: apenas 8 pontos estão representados no gráfico devido à escala dos eixos; os outros 10 pontos foram inferiores a 70%. As imagens deste tipo de chave têm grande exposição ao contexto geral do ambiente, o que pode estar levando o modelo a dar uma resposta errada devido à presença de elementos de fundo (Figura 57), especialmente após apresentar amostras reais no treinamento.

Figura 57 – Alguns casos de erro no reconhecimento de manobras de seccionadoras semi-pantográficas verticais. O corpo da chave ocupa uma região pequena da imagem, às vezes com pouco contraste em relação ao fundo.
Fonte: Autoria própria (2024).

Em resumo, a adição de dados reais melhorou as respostas dos modelos para o caso normal (chaves em seu estado usual na subestação, seja aberto ou fechado). Entretanto, para as seccionadoras de dupla abertura, observou-se redução na acurácia dos modelos para manobras (chaves no estado oposto ao usual), possivelmente devido ao modelo reconhecer características que só existem nas fotografias, associando-as ao estado usual da chave.

5.6 Discussão sobre casos de erro

Ao observar as principais amostras que os modelos classificaram incorretamente, alguns padrões podem ser encontrados. Na Figura 58, são exibidos exemplos de amostras com sérios problemas de visibilidade, seja por eventos meteorológicos ou de iluminação (natural e artificial). Em alguns casos, a chave praticamente desaparece da imagem. Essa é uma limitação do reconhecimento por imagem em um ambiente externo.

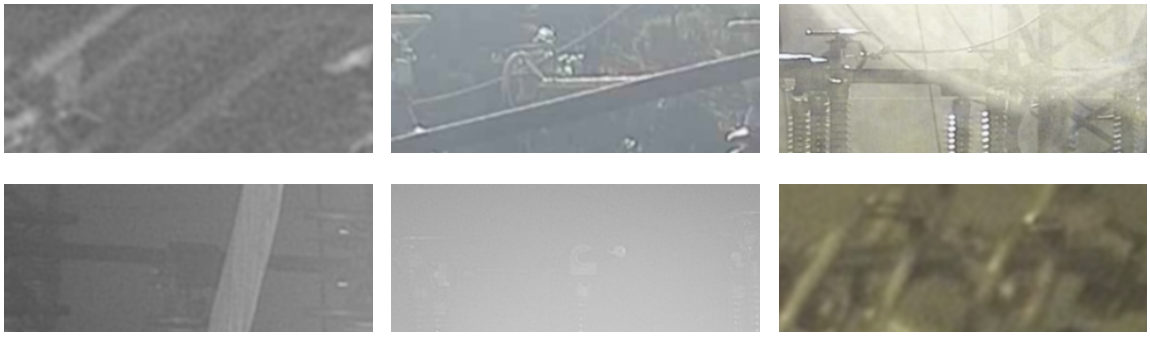


Figura 58 – Alguns casos de erro em que as amostras foram afetadas por problemas de visibilidade, tais como forte neblina, chuva, iluminação fraca ou muito intensa.
Fonte: Autoria própria (2024).

Outros erros (Figura 59) envolvem enquadramentos nos quais a chave não está em uma posição aproximadamente horizontal em relação à câmera. Por conta disso, o recorte retangular se torna excessivamente angulado e aproximado, resultando em uma amostra de baixa qualidade. Os casos mais severos já haviam sido excluídos na etapa de rotulagem (seção 4.5), mas os testes mostraram que outros enquadramentos, que estavam na margem de aceitação, possivelmente também deveriam ter sido removidos.

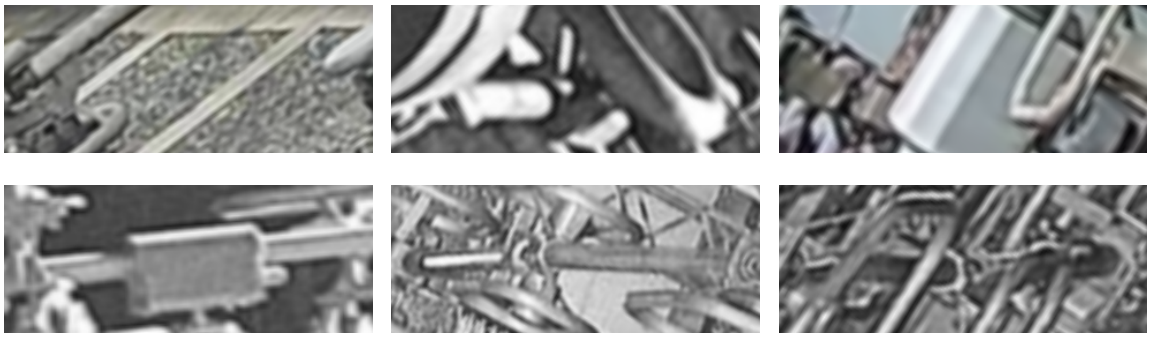


Figura 59 – Alguns casos de erro em enquadramentos com ponto de vista muito desfavorável. Sem observar o restante da imagem, estas amostras são de difícil compreensão até por um humano.
Fonte: Autoria própria (2024).

Além disso, há enquadramentos que são visualmente ambíguos. Isso ocorre principalmente em alguns enquadramentos de seccionadoras de dupla abertura, cujas projeções em 2D no estado aberto e no estado fechado têm apenas diferenças muito sutis (Figura 60), embora o corpo da chave esteja em posições totalmente diferentes no ambiente real. Outros casos de ambiguidade, ilustrados na Figura 61, incluem a confusão com objetos horizontais em primeiro plano (como cabos e estruturas metálicas) ou no fundo (como calçadas, canaletas e, também, estruturas metálicas).

É importante lembrar que, além dos casos excepcionais comentados, há também uma pequena quantidade de erros em imagens com boas condições, tais como as ilustradas na Figura 62, o que é característico de soluções baseadas em métodos de aprendizado de máquina em um conjunto de dados limitado. Entretanto, pode-se afirmar que o desempenho dos modelos

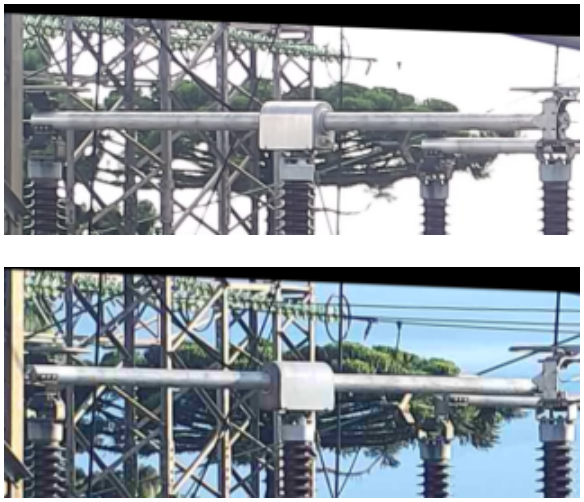


Figura 60 – Exemplo no qual a imagem da chave fechada (acima) é muito similar à imagem da chave aberta (abaixo). Essa ambiguidade visual se deve ao fato de que a seccionadora de dupla abertura gira aproximadamente 90 graus ao abrir, mas a posição relativa entre a chave e a câmera faz com que a projeção em 2D do corpo da chave apareça aproximadamente no mesmo lugar.

Fonte: Autoria própria (2024).

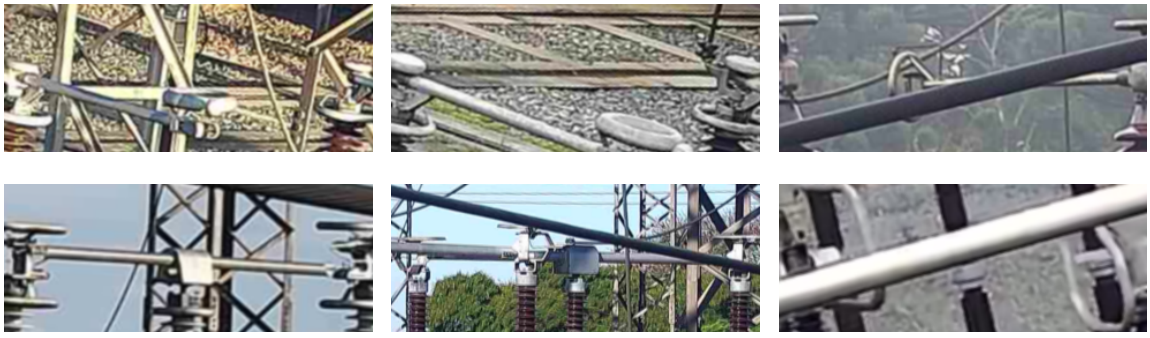


Figura 61 – Outros casos de erro relacionados à ambiguidade visual: chaves abertas que parecem fechadas, cabos e barras metálicas em primeiro plano e objetos horizontais ao fundo próximos à região da chave.

Fonte: Autoria própria (2024).

treinados como um todo é satisfatório, visto que a grande maioria dos casos foi classificada corretamente.

Figura 62 – Exemplos de imagens com boas condições de visibilidade que tiveram alguns erros de classificação. Nem todos os modelos classificaram estas imagens incorretamente: entre os 18 treinamentos (3 repetições com 6 arquiteturas distintas), a primeira imagem teve 5 erros, a segunda, 6, e a terceira, 10.

Fonte: Autoria própria (2024).

5.7 Implementação e teste em sistema embarcado

Como a plataforma NVIDIA Jetson Nano (Figura 63) possui suporte para as bibliotecas TensorFlow e Keras, a implementação do *software* para o sistema embarcado consistiu, em um primeiro momento, na adaptação dos códigos-fonte Python já desenvolvidos para o treinamento e teste, a fim de fazer as redes neurais atuarem em modo de inferência. Inicialmente, executou-se a inferência com cada rede neural de forma individual, exceto as duas redes construídas neste trabalho como *baseline* (simples e simples expandida), a fim de contabilizar os tempos de execução por amostra. Além disso, avaliou-se o tempo de inicialização do *script* (que envolve o carregamento do interpretador Python e das bibliotecas, além da alocação de memória para o modelo) e o tempo de carregamento dos pesos do modelo a partir de um arquivo. Os valores encontrados são apresentados na Tabela 11.

Figura 63 – *Kit* de desenvolvimento NVIDIA Jetson Nano
Fonte: NVIDIA Corporation[4].

Tabela 11 – Tempos médios obtidos na execução da inferência no sistema embarcado com cada rede neural testada.

Arquitetura	Inicialização (s)	Carregamento de pesos (s)	Inferência por amostra (s)
Totalmente conectada	10,8	0,2	0,07
MobileNetV2	19,6	0,5	0,15
EfficientNetB0	24,5	0,7	0,27
DenseNet-121	40,1	1,2	0,40

Fonte: Autoria própria (2024).

Para todas as arquiteturas, o tempo de execução da inferência por amostra foi considerado suficiente para a aplicação proposta, já que, durante a situação cotidiana da subestação, o sistema apenas precisa realizar a inferência em intervalos bastante espaçados (por exemplo, na ordem de 30 minutos), pois não se espera ter alteração no estado das seccionadoras. Nas

[4] https://developer.nvidia.com/embedded/jetson-nano-developer-kit

situações esporádicas em que há procedimentos de manobra, o sistema pode ser configurado para realizar a captura e inferência das imagens mais frequentemente para alguns grupos de chaves, ou imediatamente por meio do comando de um operador.

Observou-se também que o tempo de inicialização do *script* em todos os casos foi bastante elevado. Como, após a inicialização, o modelo permanece carregado e pode ser utilizado para processar múltiplas amostras, idealmente todos os modelos deveriam ser mantidos carregados em memória. Contudo, não foi possível carregar sequer duas redes neurais simultaneamente devido à limitação da plataforma embarcada, que possui apenas 4 GB de memória, a qual é compartilhada entre a CPU e a GPU. É possível aumentar a memória do sistema com o recurso de memória virtual (*swap*), que utiliza parte do espaço de armazenamento como memória adicional, mas essa solução torna o acesso à memória consideravelmente mais lento, além de acelerar a degradação da vida útil do *hardware* de armazenamento (*flash*).

Por outro lado, o tempo de carregamento de pesos, de forma geral, é equiparável a aproximadamente três inferências, sendo bastante inferior ao tempo de inicialização. Assim, mesmo com a limitação de manter apenas uma rede neural carregada por vez, é possível alterar seu conjunto de pesos rapidamente. Como foram treinados modelos (conjuntos de pesos) diferentes para cada categoria de chave, mas utilizando as mesmas arquiteturas, uma única rede neural pode ser utilizada para processar imagens de todos os tipos de chaves, mediante alteração dos pesos.

Em conclusão, a plataforma embarcada NVIDIA Jetson Nano se mostrou capaz de executar o reconhecimento do estado de chaves seccionadoras em tempo hábil, mas foi necessário otimizar o *software* para alocar apenas uma rede convolucional e alterar seu conjunto de pesos para processar imagens de tipos de chaves diferentes. Essa limitação leva à necessidade de escolher apenas uma entre as seis arquiteturas testadas, a qual não necessariamente obteve os melhores resultados nos testes para todas as categorias de chave. Para suportar modelos de arquiteturas diferentes, seria necessário desalocar a memória referente à rede neural anterior para alocar a nova, o que consumiria o tempo de inicialização novamente e impactaria significativamente o tempo do reconhecimento.

6 CONCLUSÃO

Neste trabalho, explorou-se o treinamento de redes convolucionais utilizando dados sintéticos, uma abordagem que vem ganhando destaque na atualidade devido à dificuldade na obtenção de dados reais para formar um *dataset* representativo em alguns domínios. Para estudar esse tema, considerou-se uma aplicação particular: o reconhecimento por imagem do estado de chaves seccionadoras em subestações de transmissão, que está sujeito a diversas limitações na obtenção de dados. Além de estarem em ambiente aberto e com restrições de acesso, as chaves seccionadoras normalmente permanecem por muito tempo no mesmo estado, fazendo com que coletas de dados reais normalmente tenham amostras de cada exemplar de chave somente em seu estado usual, com atividades de manobra sendo pouco frequentes e exigindo autorizações especiais mesmo para serem realizadas por um curto período.

Para abordar o problema, considerou-se a geração de amostras sintéticas através da renderização de modelos tridimensionais de subestações. Tais modelos permitem a manipulação de parâmetros do ambiente, permitindo formar conjuntos de dados extremamente variados. Entretanto, como dados sintéticos são gerados por uma simplificação da realidade, há uma distanciação entre o domínio sintético e o real, denominada *domain gap*, que pode comprometer o desempenho de um modelo. Para permitir a transferência do aprendizado em dados sintéticos para um domínio real, são empregadas diversas técnicas, tais como: randomização de domínio, composição 2D com imagens reais, uso de uma pequena quantidade de dados reais no treinamento, pesos pré-treinados em grandes conjuntos de dados, entre outras.

Nesse contexto, o presente trabalho teve o objetivo principal de determinar a viabilidade do uso de dados sintéticos para o treinamento de um sistema de aprendizado de máquinas para reconhecimento do estado de chaves seccionadoras em subestações de transmissão. A solução proposta foi aplicada em duas subestações no estado do Paraná, cujos modelos 3D foram construídos especificamente para esta finalidade. Para avaliação dos modelos treinados sobre dados sintéticos, foram instaladas câmeras físicas do tipo PTZ nas subestações. Também foi realizada uma quantidade limitada de manobras de chaves seccionadoras para coletar dados das chaves no estado oposto ao usual, a fim de verificar a capacidade de reconhecimento de mudanças de estado. Nos testes realizados no presente trabalho, os conjuntos de treinamento somam aproximadamente 1,15 milhão de amostras sintéticas e 6 mil amostras reais, enquanto os conjuntos de teste possuem no total cerca de 62 mil amostras reais, dispersas entre cinco tipos diferentes de chave.

Os experimentos envolveram três arquiteturas de redes convolucionais próprias, incluindo duas arquiteturas *baseline* simples e uma rede totalmente convolucional; além de três arquiteturas bem-conhecidas: EfficientNetB0, DenseNet-121 e MobileNetV2, inicializadas com pesos pré-treinados no conjunto ImageNet. Os treinamentos foram realizados para cada tipo de chave separadamente, ou seja, cada modelo treinado reconhece imagens de apenas um tipo de chave. Em um primeiro momento, avaliou-se a evolução do desempenho dos modelos treinados

em função da quantidade de amostras sintéticas de treinamento. De forma geral, observou-se que a taxa de acerto normalmente aumenta quando são acrescentadas amostras ao conjunto de treinamento, embora esse aumento desacelere à medida que o *dataset* cresce. Também se observou que alguns tipos de chaves são mais difíceis de reconhecer que outros: no caso da abertura central, cerca de 1.000 amostras sintéticas já produziram modelos com acurácia superior a 90%, enquanto que no caso da dupla abertura, o melhor valor alcançado no experimento inicial foi cerca de 83%, utilizando 80.000 amostras.

Em seguida, foi feita uma comparação entre modelos treinados com 150.000 amostras sintéticas, considerando diferentes tipos de chave e diferentes redes convolucionais. Os resultados mostraram a existência de um *domain gap*: acurácia de validação elevada (acima de 98% sobre dados sintéticos) e menores taxas de acerto nos testes (com melhores resultados entre 87% e 97%, a depender do tipo de chave). Para uma categoria específica de chave (semi-pantográfica vertical), o reconhecimento de manobras (chaves no estado oposto ao usual) não teve bons resultados; entretanto, para os outros quatro tipos de seccionadora, o desempenho no reconhecimento de manobras foi superior ao reconhecimento do estado das chaves em seus estados usuais.

Buscando ultrapassar a barreira do *domain gap*, avaliou-se uma estratégia de treinamento conjunto entre os dados sintéticos e uma pequena parcela de dados reais. O efeito observado foi uma melhora dos resultados nos testes com imagens das chaves em seu estado usual na subestação: os melhores modelos para cada tipo de chave alcançaram taxas de acerto superiores a 97%. Essa melhora foi mais expressiva para as chaves de dupla abertura, que tinham se mostrado mais desafiadoras no experimento anterior; entretanto, o reconhecimento de manobras desse tipo de chave teve uma pequena queda de desempenho após a adição de dados reais. Para as demais seccionadoras, o reconhecimento de manobras permaneceu estável entre os dois experimentos. Nesse experimento, as três arquiteturas bem-conhecidas tiveram bom desempenho, com destaque à DenseNet-121, cujos resultados se mantiveram mais consistentes entre os cinco tipos de chave.

Por fim, implementou-se um *software* de inferência para a plataforma embarcada NVIDIA Jetson Nano, utilizando os modelos treinados. Todas as arquiteturas testadas se mostraram viáveis para execução nessa plataforma — a arquitetura que teve o tempo de inferência por amostra mais lento foi a DenseNet-121 (0,40 segundos). Contudo, a memória do sistema não foi suficiente para manter mais de uma rede convolucional alocada simultaneamente. A saída adotada foi escolher uma única arquitetura, substituindo apenas o conjunto de pesos conforme o tipo de chave da imagem a ser processada.

Em conclusão, a utilização de dados sintéticos para treinamento de modelos de classificação, especificamente para o reconhecimento do estado de seccionadoras em subestações, se mostrou promissora. Essa abordagem foi especialmente eficaz para o reconhecimento de casos raros (manobras), para os quais não haveria imagens suficientes para treinamento apenas com dados reais. O *domain gap*, embora ainda existente, foi mitigado com a adição de uma

pequena quantidade de dados reais no treinamento. Os melhores resultados para cada tipo de chave, em sua maioria, superaram os trabalhos anteriores, sendo testados sobre conjuntos de teste mais extensos e sob condições ainda mais extremas, com variedade de pontos de vista, objetos confusos ao fundo da imagem, variações de iluminação, etc.

Para aplicar esta solução a outras subestações, novos modelos devem ser treinados sobre imagens (sintéticas e reais) das chaves das novas subestações, já que cada modelo reconhece o estado de apenas um tipo de chave em pontos de vista predefinidos. Portanto, a metodologia descrita nesta pesquisa deve ser aplicada desde o início, incluindo o mapeamento LiDAR, modelagem 3D, escolha de pontos para instalação de câmeras, definição de *presets*, etc. Futuramente, com a disponibilidade de dados de muitas subestações e maior variedade de tipos de chave, um modelo genérico poderia ser avaliado.

Trabalhos futuros podem explorar a inclusão de mais pontos de vista sintéticos, não necessariamente congruentes a pontos de vista reais, para ampliar a variabilidade dos dados de treinamento, buscando tornar o modelo invariante ao ponto de vista da chave. Também há espaço para a verificação do comportamento interno das redes neurais treinadas com dados sintéticos, analisando a ativação das suas camadas durante a inferência, a fim de obter maior compreensão sobre os casos em que a inferência não foi bem sucedida. Outra oportunidade é o uso de métodos mais sofisticados para aproximar o domínio real e o sintético, como a adaptação de domínio com modelos generativos. Do ponto de vista de arquitetura, trabalhos futuros também poderiam explorar modelos de *vision transformers*, que atualmente são o estado da arte em visão computacional. No caso dos enquadramentos para os quais houve maior dificuldade no reconhecimento, possivelmente outras abordagens de solução seriam mais adequadas, tais como a detecção de anomalia com imagens próximas aos contatos.

REFERÊNCIAS

AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA. **Regras dos Serviços de Transmissão de Energia Elétrica**: Módulo 4 — Prestação dos serviços. Brasília, 2020. Disponível em: https://www2.aneel.gov.br/cedoc/aren2020905_2_3.pdf.

ANDREWS, G. **What Is Synthetic Data?** NVIDIA, 2021. Disponível em: https://blogs.nvidia.com/blog/2021/06/08/what-is-synthetic-data/.

ARISTÓTELES. **Metafísica**. Tradução: Giovanni Reale. São Paulo: Loyola, 2002. ISBN 85-15-02427-6.

BALLARD, D. H. Generalizing the Hough transform to detect arbitrary shapes. **Pattern Recognition**, v. 13, n. 2, p. 111–122, 1981. ISSN 0031-3203. Disponível em: https://doi.org/10.1016/0031-3203(81)90009-1.

BERGER, M.; TAGLIASACCHI, A.; SEVERSKY, L.; ALLIEZ, P.; LEVINE, J.; SHARF, A.; SILVA, C. State of the art in surface reconstruction from point clouds. *In*: **Eurographics 2014 - State of the Art Reports**. Strasbourg, France: [*s.n.*], 2014. (EUROGRAPHICS star report, v. 1), p. 161–185. Disponível em: https://doi.org/10.2312/egst.20141040.

BORKMAN, S.; CRESPI, A.; DHAKAD, S.; GANGULY, S.; HOGINS, J.; JHANG, Y.; KAMALZADEH, M.; LI, B.; LEAL, S.; PARISI, P.; ROMERO, C.; SMITH, W.; THAMAN, A.; WARREN, S.; YADAV, N. **Unity Perception: Generate Synthetic Data for Computer Vision**. arXiv, 2021. Disponível em: https://doi.org/10.48550/ARXIV.2107.04259.

CHEN, H.; ZHAO, X.; TAN, M.; SUN, S. Computer vision-based detection and state recognition for disconnecting switch in substation automation. **International Journal of Robotics and Automation**, v. 32, p. 1–12, 2017. Disponível em: https://doi.org/10.2316/Journal.206.2017.1.206-4624.

CHEN, J.; WU, G.; JIAN, X.; CAI, L.; CHEN, S.; CHEN, R. Research on state monitoring system of intelligent disconnecting switch based on sensing technology. **Frontiers in Energy Research**, v. 10, 2022. ISSN 2296-598X. Disponível em: https://doi.org/10.3389/fenrg.2022.895301.

CHEN, L.; WAN, S.; DOU, L. Improving diagnostic performance of high-voltage circuit breakers on imbalanced data using an oversampling method. **IEEE Transactions on Power Delivery**, v. 37, n. 4, p. 2704–2716, 2022. Disponível em: https://doi.org/10.1109/TPWRD.2021.3114547.

CHEN, Y.; LI, W.; CHEN, X.; GOOL, L. V. Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. *In*: **2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [*s.n.*], 2019. p. 1841–1850. Disponível em: https://doi.org/10.1109/CVPR.2019.00194.

CORDTS, M.; OMRAN, M.; RAMOS, S.; REHFELD, T.; ENZWEILER, M.; BENENSON, R.; FRANKE, U.; ROTH, S.; SCHIELE, B. The Cityscapes dataset for semantic urban scene understanding. *In*: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*s.n.*], 2016. Disponível em: https://doi.org/10.1109/CVPR.2016.350.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine Learning**, v. 20, n. 3, p. 273–297, Sep 1995. ISSN 1573-0565. Disponível em: https://doi.org/10.1007/BF00994018.

CSURKA, G. A comprehensive survey on domain adaptation for visual applications. *In*: ____. **Domain Adaptation in Computer Vision Applications**. Cham: Springer International Publishing, 2017. p. 1–35. ISBN 978-3-319-58347-1. Disponível em: https://doi.org/10.1007/978-3-319-58347-1_1.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. *In*: **2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)**. [*s.n.*], 2005. v. 1, p. 886–893 vol. 1. Disponível em: https://doi.org/10.1109/CVPR.2005.177.

DAVARI, N.; AKBARIZADEH, G.; MASHHOUR, E. Corona detection and power equipment classification based on GoogleNet-AlexNet: An accurate and intelligent defect detection model based on deep learning for power distribution lines. **IEEE Transactions on Power Delivery**, v. 37, n. 4, p. 2766–2774, 2022. Disponível em: https://doi.org/10.1109/TPWRD.2021.3116489.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A large-scale hierarchical image database. *In*: **2009 IEEE Conference on Computer Vision and Pattern Recognition**. [*s.n.*], 2009. p. 248–255. Disponível em: https://doi.org/10.1109/CVPR.2009.5206848.

EBADI, S. E.; JHANG, Y.; ZOOK, A.; DHAKAD, S.; CRESPI, A.; PARISI, P.; BORKMAN, S.; HOGINS, J.; GANGULY, S. **PeopleSansPeople: A Synthetic Data Generator for Human-Centric Computer Vision**. arXiv, 2021. Disponível em: https://doi.org/10.48550/ARXIV.2112.09290.

EVERINGHAM, M.; GOOL, L. V.; WILLIAMS, C. K. I.; WINN, J.; ZISSERMAN, A. The PASCAL Visual Object classes (VOC) Challenge. **International Journal of Computer Vision**, v. 88, n. 2, p. 303–338, jun 2010. Disponível em: https://doi.org/10.1007/s11263-009-0275-4.

FOROOSH, H.; ZERUBIA, J.; BERTHOD, M. Extension of phase correlation to subpixel registration. **IEEE Transactions on Image Processing**, v. 11, n. 3, p. 188–200, 2002. Disponível em: https://doi.org/10.1109/83.988953.

FOWLER, D. R.; MEINHARDT, H.; PRUSINKIEWICZ, P. Modeling seashells. **SIGGRAPH Comput. Graph.**, Association for Computing Machinery, New York, NY, USA, v. 26, n. 2, p. 379–387, jul 1992. ISSN 0097-8930. Disponível em: https://doi.org/10.1145/142920.134096.

FU, J.; SUN, R.; XU, Z.; ZHU, J.; CHEN, Y.; GAO, B. Substation isolation switch state recognition technology based on image line segment fitting. *In*: **2019 IEEE 9th Annual International Conference on CYBER Technology in Automation, Control, and Intelligent Systems (CYBER)**. [*s.n.*], 2019. p. 1625–1629. Disponível em: https://doi.org/10.1109/CYBER46603.2019.9066752.

GAIDON, A.; WANG, Q.; CABON, Y.; VIG, E. Virtual worlds as proxy for multi-object tracking analysis. *In*: **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Los Alamitos, CA, USA: IEEE Computer Society, 2016. p. 4340–4349. ISSN 1063-6919. Disponível em: https://doi.org/10.1109/CVPR.2016.470.

GEIGER, A.; LENZ, P.; STILLER, C.; URTASUN, R. Vision meets robotics: The KITTI dataset. **The International Journal of Robotics Research (IJRR)**, v. 32, n. 11, p. 1231–1237, 2013. Disponível em: https://doi.org/10.1177/0278364913491297.

GOMES, V. B.; MARCHESI, B.; GRUBER, Y. A.; LIPPMANN JR, L.; SCREMIN, M.; NASSU, B. T. **Exploring Synthetic Data for Training Deep Learning Models for High-Voltage Disconnector State Identification**. 2024. Submetido para revisão.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. MIT Press, 2016. https://www.deeplearningbook.org. Disponível em: https://www.deeplearningbook.org.

GRETTON, A.; BORGWARDT, K. M.; RASCH, M. J.; SCHöLKOPF, B.; SMOLA, A. A kernel two-sample test. **Journal of Machine Learning Research**, v. 13, n. 25, p. 723–773, 2012. Disponível em: http://jmlr.org/papers/v13/gretton12a.html.

GRUEN, A.; REMONDINO, F.; ZHANG, L. Photogrammetric reconstruction of the Great Buddha of Bamiyan, Afghanistan. **The Photogrammetric Record**, v. 19, p. 177 – 199, 09 2004. Disponível em: https://doi.org/10.1111/j.0031-868X.2004.00278.x.

HAN, S.; YANG, F.; JIANG, H.; YANG, G.; ZHANG, N.; WANG, D. A smart thermography camera and application in the diagnosis of electrical equipment. **IEEE Transactions on Instrumentation and Measurement**, v. 70, p. 1–8, 2021. Disponível em: https://doi.org/10.1109/TIM.2021.3094235.

HAN, X.; ZHANG, Z.; DING, N.; GU, Y.; LIU, X.; HUO, Y.; QIU, J.; YAO, Y.; ZHANG, A.; ZHANG, L.; HAN, W.; HUANG, M.; JIN, Q.; LAN, Y.; LIU, Y.; LIU, Z.; LU, Z.; QIU, X.; SONG, R.; TANG, J.; WEN, J.-R.; YUAN, J.; ZHAO, W. X.; ZHU, J. Pre-trained models: Past, present and future. **AI Open**, v. 2, p. 225–250, 2021. ISSN 2666-6510. Disponível em: https://doi.org/10.1016/j.aiopen.2021.08.002.

HINTERSTOISSER, S.; LEPETIT, V.; WOHLHART, P.; KONOLIGE, K. On pre-trained image features and synthetic images for deep learning. *In*: LEAL-TAIXÉ, L.; ROTH, S. (Ed.). **Computer Vision – ECCV 2018 Workshops**. Cham: Springer International Publishing, 2019. p. 682–697. ISBN 978-3-030-11009-3. Disponível em: https://doi.org/10.1007/978-3-030-11009-3_42.

HUANG, G.; LIU, Z.; MAATEN, L. V. D.; WEINBERGER, K. Q. Densely connected convolutional networks. *In*: **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*s.n.*], 2017. p. 2261–2269. Disponível em: https://doi.org/10.1109/CVPR.2017.243.

HUANG, H.; LI, D.; ZHANG, H.; ASCHER, U.; COHEN-OR, D. Consolidation of unorganized point clouds for surface reconstruction. **ACM Trans. Graph.**, Association for Computing Machinery, New York, NY, USA, v. 28, n. 5, p. 1–7, dec 2009. ISSN 0730-0301. Disponível em: https://doi.org/10.1145/1618452.1618522.

HUANG, S.; SHANG, B.; SONG, Y.; ZHANG, N.; WANG, S.; NING, S. Research on real-time disconnector state evaluation method based on multi-source images. **IEEE Transactions on Instrumentation and Measurement**, v. 71, p. 1–15, 2022. Disponível em: https://doi.org/10.1109/TIM.2021.3137864.

HUGHES, J. F.; DAM, A. van; MCGUIRE, M.; SKLAR, D. F.; FOLEY, J. D.; FEINER, S. K.; AKELEY, K. **Computer Graphics: Principles and Practice**. 3. ed. [*S.l.*]: Addison-Wesley, 2013. ISBN 978-0-321-39952-6.

IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *In*: BACH, F.; BLEI, D. (Ed.). **Proceedings of the 32nd International Conference on Machine Learning**. Lille, France: PMLR, 2015. (Proceedings of Machine Learning Research, v. 37), p. 448–456. Disponível em: https://proceedings.mlr.press/v37/ioffe15.html.

JAMES, S.; WOHLHART, P.; KALAKRISHNAN, M.; KALASHNIKOV, D.; IRPAN, A.; IBARZ, J.; LEVINE, S.; HADSELL, R.; BOUSMALIS, K. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. *In*: **2019 IEEE/CVF Conference on**

Computer Vision and Pattern Recognition (CVPR). [*s.n.*], 2019. p. 12619–12629. Disponível em: https://doi.org/10.1109/CVPR.2019.01291.

KAI, Z.; ZHAO, T.; ZHIJIAN, H.; LIXIONG, Y.; KE, D.; RUYU, B.; YULEI, L. Substation switch state recognition method based on NSST image fusion. *In*: **2021 4th International Conference on Circuits, Systems and Simulation (ICCSS)**. [*s.n.*], 2021. p. 231–238. Disponível em: https://doi.org/10.1109/ICCSS51193.2021.9464202.

KOLLURI, R.; SHEWCHUK, J. R.; O'BRIEN, J. F. Spectral surface reconstruction from noisy point clouds. *In*: **Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing**. New York, NY, USA: Association for Computing Machinery, 2004. (SGP '04), p. 11–21. ISBN 3905673134. Disponível em: https://doi.org/10.1145/1057432.1057434.

KOUTSOUDIS, A.; ARNAOUTOGLOU, F.; CHAMZAS, C. On 3D reconstruction of the old city of Xanthi. A minimum budget approach to virtual touring based on photogrammetry. **Journal of Cultural Heritage**, v. 8, n. 1, p. 26–31, 2007. ISSN 1296-2074. Disponível em: https://doi.org/10.1016/j.culher.2006.08.003.

LE, Q.-S.; PHAM, V.-C.; PHUNG, B.-L. Vision based state recognition of 220kv disconnect switches in power substations. *In*: **2022 6th International Conference on Green Technology and Sustainable Development (GTSD)**. [*s.n.*], 2022. p. 895–898. Disponível em: https://doi.org/10.1109/GTSD54989.2022.9989321.

LIN, M.; CHEN, Q.; YAN, S. Network in network. *In*: BENGIO, Y.; LECUN, Y. (Ed.). **2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings**. [*s.n.*], 2014. Disponível em: https://doi.org/10.48550/arXiv.1312.4400.

LIN, T.; MAIRE, M.; BELONGIE, S.; HAYS, J.; PERONA, P.; RAMANAN, D.; DOLLÁR, P.; ZITNICK, C. L. Microsoft COCO: Common objects in context. *In*: FLEET, D.; PAJDLA, T.; SCHIELE, B.; TUYTELAARS, T. (Ed.). **Computer Vision – ECCV 2014**. Cham: Springer International Publishing, 2014. p. 740–755. ISBN 978-3-319-10602-1. Disponível em: https://doi.org/10.1007/978-3-319-10602-1_48.

LIPPMANN JR, L.; SCREMIN, M.; MARCHESI, B.; GOMES, V. B.; NASSU, B. T.; ALMEIDA, A. C. de; GRUBER, Y. A.; WAGNER, R. Ferramenta de escolha de pontos ótimos para posicionamento de câmeras para observação de elementos estratégicos dos circuitos das subestações de transmissão. *In*: **XXVI Seminário Nacional de Produção e Transmissão de Energia Elétrica**. [*S.l.*: *s.n.*], 2022.

LIU, Y.; SHI, H.; SHEN, H.; SI, Y.; WANG, X.; MEI, T. A new dataset and boundary-attention semantic segmentation for face parsing. **Proceedings of the AAAI Conference on Artificial Intelligence**, v. 34, n. 07, p. 11637–11644, 4 2020. Disponível em: https://doi.org/10.1609/aaai.v34i07.6832.

LOWE, D. G. Distinctive image features from scale-invariant keypoints. **International Journal of Computer Vision**, v. 60, n. 2, p. 91–110, Nov 2004. ISSN 1573-1405. Disponível em: https://doi.org/10.1023/B:VISI.0000029664.99615.94.

LU, J.; LIN, H.; ZHANG, W.; SHI, X. A condition monitoring algorithm based on image geometric analysis for substation switch. *In*: **Proceedings of 2015 International Conference on Intelligent Computing and Internet of Things**. [*s.n.*], 2015. p. 72–76. Disponível em: https://doi.org/10.1109/ICAIOT.2015.7111541.

MCDONALD, J. D. **Electric Power Substations Engineering**. CRC Press, 2017. (Electrical Engineering Handbook). ISBN 9781439856390. Disponível em: https://books.google.com.br/books?id=RynOBQAAQBAJ.

MEHTA, B.; DIAZ, M.; GOLEMO, F.; PAL, C. J.; PAULL, L. Active Domain Randomization. *In*: KAELBLING, L. P.; KRAGIC, D.; SUGIURA, K. (Ed.). **Proceedings of the Conference on Robot Learning**. PMLR, 2020. (Proceedings of Machine Learning Research, v. 100), p. 1162–1176. Disponível em: https://proceedings.mlr.press/v100/mehta20a.html.

MOGOS, A. S.; LIANG, X.; CHUNG, C. Y. Distribution transformer failure prediction for predictive maintenance using hybrid one-class deep SVDD classification and lightning strike failures data. **IEEE Transactions on Power Delivery**, v. 38, n. 5, p. 3250–3261, 2023. Disponível em: https://doi.org/10.1109/TPWRD.2023.3268248.

MOK, R. W.; SILVEIRA, P.; DANTE, A.; VARGAS, J.; AO, T. T.; BELLINI, R.; CARVALHO, C.; ALLIL, R.; WERNECK, M. Disconnect switch position sensor based on FBG. *In*: **2019 IEEE SENSORS**. [*s.n.*], 2019. p. 1–4. Disponível em: https://doi.org/10.1109/SENSORS43011.2019.8956678.

NASSU, B. T.; MARCHESI, B.; WAGNER, R.; GOMES, V. B.; ZARNICINSKI, V.; LIPPMANN, L. A computer vision system for monitoring disconnect switches in distribution substations. **IEEE Transactions on Power Delivery**, v. 37, n. 2, p. 833–841, 2022. Disponível em: https://doi.org/10.1109/TPWRD.2021.3071971.

NIKOLENKO, S. I. **Synthetic Data for Deep Learning**. Springer International Publishing, 2021. ISBN 978-3-030-75178-4. Disponível em: https://doi.org/10.1007/978-3-030-75178-4.

NOWRUZI, F. E.; KAPOOR, P.; KOLHATKAR, D.; HASSANAT, F. A.; LAGANIERE, R.; REBUT, J. **How much real data do we actually need: Analyzing object detection performance using synthetic and real data**. arXiv, 2019. Disponível em: https://doi.org/10.48550/ARXIV.1907.07061.

OLIVEIRA, M. M. Image-based modeling and rendering techniques: A survey. **Revista de Informática Teórica e Aplicada**, v. 9, p. 37–66, 01 2002.

O'MAHONY, N.; CAMPBELL, S.; CARVALHO, A.; HARAPANAHALLI, S.; HERNANDEZ, G. V.; KRPALKOVA, L.; RIORDAN, D.; WALSH, J. Deep learning vs. traditional computer vision. *In*: ARAI, K.; KAPOOR, S. (Ed.). **Advances in Computer Vision**. Cham: Springer International Publishing, 2020. p. 128–144. ISBN 978-3-030-17795-9. Disponível em: https://doi.org/10.1007/978-3-030-17795-9_10.

OU, J.; WANG, J.; XUE, J.; WANG, J.; ZHOU, X.; SHE, L.; FAN, Y. Infrared image target detection of substation electrical equipment using an improved Faster R-CNN. **IEEE Transactions on Power Delivery**, v. 38, n. 1, p. 387–396, 2023. Disponível em: https://doi.org/10.1109/TPWRD.2022.3191694.

PHARR, M.; JAKOB, W.; HUMPHREYS, G. **Physically Based Rendering: From Theory to Implementation**. 3. ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2016. ISBN 0128006455. Disponível em: https://www.pbr-book.org/.

PORTER, T.; DUFF, T. Compositing digital images. *In*: **Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques**. New York, NY, USA: Association for Computing Machinery, 1984. (SIGGRAPH '84), p. 253–259. ISBN 0897911385. Disponível em: https://doi.org/10.1145/800031.808606.

PRAKASH, A.; BOOCHOON, S.; BROPHY, M.; ACUNA, D.; CAMERACCI, E.; STATE, G.; SHAPIRA, O.; BIRCHFIELD, S. Structured Domain Randomization: Bridging the reality gap by context-aware synthetic data. *In*: **2019 International Conference on Robotics and Automation (ICRA)**. [*s.n.*], 2019. p. 7249–7255. Disponível em: https://doi.org/10.1109/ICRA.2019.8794443.

QUAN, W.; FENG, K.; LU, X.; LIN, G.; LIU, X.; XIANG, M.; GU, G. A state recognition method of isolation switch in traction substation based on key components detection and geometric ranging. **Neural Processing Letters**, v. 54, n. 6, p. 5565–5585, 2022. Disponível em: https://doi.org/10.1007/s11063-022-10874-x.

RICHTER, S. R.; VINEET, V.; ROTH, S.; KOLTUN, V. Playing for data: Ground truth from computer games. *In*: LEIBE, B.; MATAS, J.; SEBE, N.; WELLING, M. (Ed.). **Computer Vision – ECCV 2016**. Cham: Springer International Publishing, 2016. p. 102–118. ISBN 978-3-319-46475-6. Disponível em: https://doi.org/10.1007/978-3-319-46475-6_7.

ROS, G.; SELLART, L.; MATERZYNSKA, J.; VAZQUEZ, D.; LOPEZ, A. M. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. *In*: **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*s.n.*], 2016. Disponível em: https://doi.org/10.1109/CVPR.2016.352.

RUSSAKOVSKY, O.; DENG, J.; SU, H.; KRAUSE, J.; SATHEESH, S.; MA, S.; HUANG, Z.; KARPATHY, A.; KHOSLA, A.; BERNSTEIN, M.; BERG, A. C.; FEI-FEI, L. ImageNet Large Scale Visual Recognition Challenge. **International Journal of Computer Vision (IJCV)**, v. 115, n. 3, p. 211–252, 2015. Disponível em: https://doi.org/10.1007/s11263-015-0816-y.

SAGONAS, C.; ANTONAKOS, E.; TZIMIROPOULOS, G.; ZAFEIRIOU, S.; PANTIC, M. 300 faces in-the-wild challenge: database and results. **Image and Vision Computing**, v. 47, p. 3–18, 2016. ISSN 0262-8856. 300-W, the First Automatic Facial Landmark Detection in-the-Wild Challenge. Disponível em: https://doi.org/10.1016/j.imavis.2016.01.002.

SANDLER, M.; HOWARD, A.; ZHU, M.; ZHMOGINOV, A.; CHEN, L.-C. MobileNetV2: Inverted residuals and linear bottlenecks. *In*: **2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [*s.n.*], 2018. p. 4510–4520. Disponível em: https://doi.org/10.1109/CVPR.2018.00474.

SANKARANARAYANAN, S.; BALAJI, Y.; JAIN, A.; LIM, S. N.; CHELLAPPA, R. Learning from synthetic data: Addressing domain shift for semantic segmentation. *In*: **2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition**. [*s.n.*], 2018. p. 3752–3761. Disponível em: https://doi.org/10.1109/CVPR.2018.00395.

SEMEDO, S. M. V.; OLIVEIRA, J. E. G.; CARDOSO, F. J. A. Remote monitoring of high-voltage disconnect switches in electrical distribution substations. *In*: **2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)**. [*s.n.*], 2014. p. 2060–2064. Disponível em: https://doi.org/10.1109/ISIE.2014.6864934.

SHAN, J.; TOTH, C. **Topographic Laser Ranging and Scanning: Principles and Processing**. 2. ed. CRC Press, 2018. ISBN 9781351650427. Disponível em: https://books.google.com.br/books?id=N_ErDwAAQBAJ.

SHORTEN, C.; KHOSHGOFTAAR, T. M. A survey on image data augmentation for deep learning. **Journal of Big Data**, v. 6, p. 1–48, 2019. Disponível em: https://doi.org/10.1186/s40537-019-0197-0.

SPRINGENBERG, J. T.; DOSOVITSKIY, A.; BROX, T.; RIEDMILLER, M. A. Striving for simplicity: The all convolutional net. *In*: BENGIO, Y.; LECUN, Y. (Ed.). **3rd International Conference on**

Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings. [*s.n.*], 2015. Disponível em: https://doi.org/10.48550/arXiv.1412.6806.

STRICKLAND, E. Andrew Ng, AI minimalist: The machine-learning pioneer says small is the new big. **IEEE Spectrum**, v. 59, n. 4, p. 22–50, 2022. Disponível em: https://doi.org/10.1109/MSPEC.2022.9754503.

SZEGEDY, C.; LIU, W.; JIA, Y.; SERMANET, P.; REED, S.; ANGUELOV, D.; ERHAN, D.; VANHOUCKE, V.; RABINOVICH, A. Going deeper with convolutions. *In*: **2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*s.n.*], 2015. p. 1–9. Disponível em: https://doi.org/10.1109/CVPR.2015.7298594.

SZEGEDY, C.; VANHOUCKE, V.; IOFFE, S.; SHLENS, J.; WOJNA, Z. Rethinking the Inception architecture for computer vision. *In*: **2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [*s.n.*], 2016. p. 2818–2826. Disponível em: https://doi.org/10.1109/CVPR.2016.308.

TAN, M.; LE, Q. EfficientNet: Rethinking model scaling for convolutional neural networks. *In*: CHAUDHURI, K.; SALAKHUTDINOV, R. (Ed.). **Proceedings of the 36th International Conference on Machine Learning**. PMLR, 2019. (Proceedings of Machine Learning Research, v. 97), p. 6105–6114. Disponível em: https://proceedings.mlr.press/v97/tan19a.html.

TENG, Y.; TAN, T.-Y.; LEI, C.; YANG, J.-G.; MA, Y.; ZHAO, K.; JIA, Y.-Y.; LIU, Y. A novel method to recognize the state of high-voltage isolating switch. **IEEE Transactions on Power Delivery**, v. 34, n. 4, p. 1350–1356, 2019. Disponível em: https://doi.org/10.1109/TPWRD.2019.2897132.

TOBIN, J.; FONG, R.; RAY, A.; SCHNEIDER, J.; ZAREMBA, W.; ABBEEL, P. Domain randomization for transferring deep neural networks from simulation to the real world. *In*: **2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)**. [*s.n.*], 2017. p. 23–30. Disponível em: https://doi.org/10.1109/IROS.2017.8202133.

TREMBLAY, J.; PRAKASH, A.; ACUNA, D.; BROPHY, M.; JAMPANI, V.; ANIL, C.; TO, T.; CAMERACCI, E.; BOOCHOON, S.; BIRCHFIELD, S. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. *In*: **2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)**. [*s.n.*], 2018. Disponível em: https://doi.org/10.1109/CVPRW.2018.00143.

TRIPATHI, S.; CHANDRA, S.; AGRAWAL, A.; TYAGI, A.; REHG, J. M.; CHARI, V. Learning to generate synthetic data via compositing. *In*: **2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)**. [*s.n.*], 2019. p. 461–470. Disponível em: https://doi.org/10.1109/CVPR.2019.00055.

WANG, J. Deep learning based state recognition of substation switches. *In*: **AIP Conference Proceedings**. [*s.n.*], 2018. v. 1971, n. 1. ISSN 0094-243X. Disponível em: https://doi.org/10.1063/1.5041183.

WANG, M.; DENG, W. Deep visual domain adaptation: A survey. **Neurocomputing**, v. 312, p. 135–153, 2018. ISSN 0925-2312. Disponível em: https://doi.org/10.1016/j.neucom.2018.05.083.

WANG, T.; TAN, Y.; WANG, Y.; JIN, B.; MONTI, A.; SANGIOVANNI-VINCENTELLI, A. L. Synthetic data in DC microgrids: Label creation for ensemble learning for fault isolation. **IEEE Transactions on Power Delivery**, v. 37, n. 3, p. 2301–2313, 2022. Disponível em: https://doi.org/10.1109/TPWRD.2021.3110182.

WANG, Y.; YAN, J.; YANG, Z.; JING, Q.; QI, Z.; WANG, J.; GENG, Y. A domain adaptive deep transfer learning method for gas-insulated switchgear partial discharge diagnosis. **IEEE Transactions on Power Delivery**, v. 37, n. 4, p. 2514–2523, 2022. Disponível em: https://doi.org/10.1109/TPWRD.2021.3111862.

WERNECK, M. M.; ALLIL, R. C. **Monitoramento remoto de chaves seccionadoras por sensores a fibra óptica**. ResearchGate, 2019. Disponível em: https://www.researchgate.net/publication/333783441_MONITORAMENTO_REMOTO_DE_CHAVES_SECCIONADORAS_POR_SENSORES_A_FIBRA_OPTICA.

WHITING, E.; OCHSENDORF, J.; DURAND, F. Procedural modeling of structurally-sound masonry buildings. *In*: **ACM SIGGRAPH Asia 2009 Papers**. New York, NY, USA: Association for Computing Machinery, 2009. (SIGGRAPH Asia '09). ISBN 9781605588582. Disponível em: https://doi.org/10.1145/1661412.1618458.

WOOD, E.; BALTRUŠAITIS, T.; HEWITT, C.; DZIADZIO, S.; CASHMAN, T. J.; SHOTTON, J. Fake it till you make it: Face analysis in the wild using synthetic data alone. *In*: **2021 IEEE/CVF International Conference on Computer Vision (ICCV)**. [*s.n.*], 2021. p. 3661–3671. Disponível em: https://doi.org/10.1109/ICCV48922.2021.00366.

YOSINSKI, J.; CLUNE, J.; BENGIO, Y.; LIPSON, H. How transferable are features in deep neural networks? *In*: GHAHRAMANI, Z.; WELLING, M.; CORTES, C.; LAWRENCE, N.; WEINBERGER, K. (Ed.). **Advances in Neural Information Processing Systems**. Curran Associates, Inc., 2014. v. 27. Disponível em: https://proceedings.neurips.cc/paper/2014/file/375c71349b295fbe2dcdca9206f20a06-Paper.pdf.

YU, F.; LU, Z.; LUO, H.; WANG, P. **Three-Dimensional Model Analysis and Processing**. Springer, 2011. ISBN 978-3-642-12650-5. Disponível em: https://doi.org/10.1007/978-3-642-12651-2.

ZHAO, Z.; FENG, S.; ZHAI, Y.; ZHAO, W.; LI, G. Infrared thermal image instance segmentation method for power substation equipment based on visual feature reasoning. **IEEE Transactions on Instrumentation and Measurement**, v. 72, p. 1–13, 2023. Disponível em: https://doi.org/10.1109/TIM.2023.3322998.

ZHENG, H.; CUI, Y.; YANG, W.; LI, J.; JI, L.; PING, Y.; HU, S.; CHEN, X. An infrared image detection method of substation equipment combining Iresgroup structure and CenterNet. **IEEE Transactions on Power Delivery**, v. 37, n. 6, p. 4757–4765, 2022. Disponível em: https://doi.org/10.1109/TPWRD.2022.3158818.

ZHOU, S.; LIU, J.; FAN, X.; FU, Q.; GOH, H. H. Thermal fault diagnosis of electrical equipment in substations using lightweight convolutional neural network. **IEEE Transactions on Instrumentation and Measurement**, v. 72, p. 1–9, 2023. Disponível em: https://doi.org/10.1109/TIM.2023.3240210.

ZITOVÁ, B.; FLUSSER, J. Image registration methods: a survey. **Image and Vision Computing**, v. 21, n. 11, p. 977–1000, 2003. ISSN 0262-8856. Disponível em: https://doi.org/10.1016/S0262-8856(03)00137-9.

APÊNDICE A – Resultados detalhados

Nas Tabelas 12 e 13, apresentam-se os resultados médios para treinamentos com 150 mil amostras sintéticas (sem e com a adição de dados reais, respectivamente), agrupados por arquitetura de rede neural convolucional e por *dataset*. Cada célula é colorida entre vermelho (80% ou abaixo) e verde (100%). Os resultados em **negrito** são os melhores de cada linha, dando ênfase ao melhor resultado obtido para um *dataset* de teste específico entre as arquiteturas testadas. Já os resultados sublinhados são os melhores de cada coluna, destacando o *dataset* no qual um determinado modelo teve melhor desempenho. Em caso de empate, mais de um resultado foi destacado. O conjunto de validação não foi considerado nesta comparação.

Tabela 12 – Acurácia dos modelos treinados apenas sobre dados sintéticos.

	Rede convolucional					
Dataset	Simples	Simples exp.	Total. conv.	MobileNetV2	EfficientNetB0	DenseNet-121
	Abertura central 1					
Validação	0,9903	0,9935	0,9949	0,9884	0,9911	0,9934
Normal 1	0,9589	0,9527	0,8753	0,7086	**0,9681**	0,9376
Normal 2	0,9501	0,9540	0,8859	0,6951	**0,9764**	0,9485
Manobras	0,9853	0,9963	0,9744	0,9927	**1,0000**	**1,0000**
	Abertura central 2					
Validação	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Normal 1	0,9746	**0,9749**	0,8601	0,9393	0,9541	0,9429
Normal 2	0,9538	0,9576	0,8197	0,9278	**0,9610**	0,9467
Manobras	0,9815	**1,0000**	0,9444	0,9815	0,9815	**1,0000**
	Dupla abertura					
Validação	0,9897	0,9928	0,9944	0,9882	0,9927	0,9923
Normal 1	0,8180	0,8014	0,8335	0,8644	**0,8777**	0,8429
Normal 2	0,8116	0,8221	0,8392	0,8748	**0,8981**	0,8568
Manobras	0,9674	**0,9819**	0,9529	0,9601	0,9674	0,9420
	Semi-pantográfica vertical					
Validação	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Normal 1	0,9973	0,9946	**1,0000**	0,9998	0,9967	0,9992
Normal 2	**1,0000**	0,9984	**1,0000**	**1,0000**	**1,0000**	0,9953
Manobras	0,5833	0,8056	0,1944	0,8333	**0,8611**	0,8611
	Semi-pantográfica horizontal					
Validação	0,9812	0,9834	0,9881	0,9831	0,9810	0,9855
Normal 1	**0,9503**	0,9476	0,9276	0,7675	0,9492	0,8469
Normal 2	0,9498	**0,9525**	0,9198	0,7491	0,9414	0,8343
Manobras	0,9647	0,9683	**0,9735**	0,9189	0,9718	0,9665

Fonte: Autoria própria (2024).

Tabela 13 – Acurácia dos modelos treinados sobre dados sintéticos com adição de dados reais.

	Rede convolucional					
Dataset	Simples	Simples exp.	Total. conv.	MobileNetV2	EfficientNetB0	DenseNet-121
	Abertura central 1					
Validação	0,9903	0,9919	0,9932	0,9880	0,9868	0,9919
Normal 1	0,9977	0,9970	0,9969	0,9950	0,9964	**0,9991**
Normal 2	0,9926	0,9932	0,9864	0,9901	0,9968	**0,9977**
Manobras	0,9744	0,9853	0,9780	**1,0000**	0,9963	**1,0000**
	Abertura central 2					
Validação	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Normal 1	0,9948	0,9984	0,9830	0,9990	**0,9995**	**0,9995**
Normal 2	0,9706	0,9830	0,9287	0,9997	0,9997	**1,0000**
Manobras	**1,0000**	**1,0000**	**1,0000**	**1,0000**	**1,0000**	**1,0000**
	Dupla abertura					
Validação	0,9892	0,9923	0,9918	0,9890	0,9761	0,9860
Normal 1	0,9289	0,9618	0,9603	**0,9906**	0,9634	0,9705
Normal 2	0,8956	0,9237	0,9402	**0,9846**	0,9606	0,9681
Manobras	0,8986	0,8696	0,9384	0,9384	0,9348	**0,9457**
	Semi-pantográfica vertical					
Validação	1,0000	1,0000	1,0000	1,0000	1,0000	1,0000
Normal 1	0,9994	**1,0000**	**1,0000**	**1,0000**	**1,0000**	**1,0000**
Normal 2	**1,0000**	**1,0000**	**1,0000**	**1,0000**	**1,0000**	**1,0000**
Manobras	0,5556	0,4444	0,1389	0,7500	0,8611	**0,9167**
	Semi-pantográfica horizontal					
Validação	0,9811	0,9831	0,9879	0,9825	0,9791	0,9853
Normal 1	0,9529	0,9458	0,9500	0,9695	**0,9781**	0,9572
Normal 2	0,9364	0,9473	0,9508	0,9695	**0,9779**	0,9505
Manobras	0,9771	0,9788	0,9718	0,9841	0,9735	**0,9947**

Fonte: Autoria própria (2024).