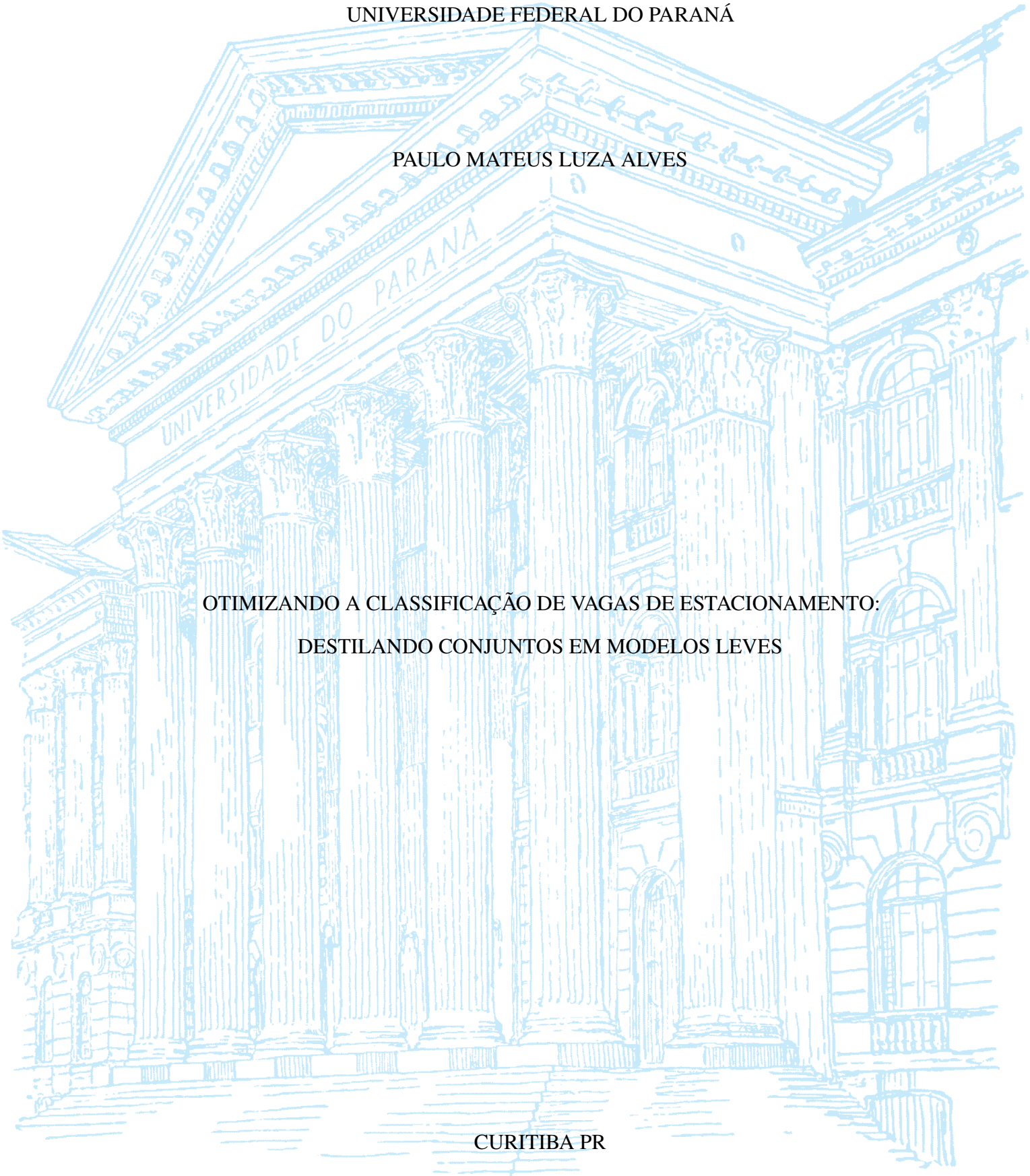
UNIVERSIDADE FEDERAL DO PARANÁ

PAULO MATEUS LUZA ALVES




# OTIMIZANDO A CLASSIFICAÇÃO DE VAGAS DE ESTACIONAMENTO: DESTILANDO CONJUNTOS EM MODELOS LEVES

CURITIBA PR

2024

PAULO MATEUS LUZA ALVES

# OTIMIZANDO A CLASSIFICAÇÃO DE VAGAS DE ESTACIONAMENTO: DESTILANDO CONJUNTOS EM MODELOS LEVES

Trabalho apresentado como requisito parcial à conclusão do Curso de Bacharelado em Ciência da Computação, Setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Computação*.

Orientador: Paulo Ricardo Lisboa de Almeida.

CURITIBA PR

2024

*A minha família, meus maiores apoiadores e companheiros de todos os dias.*

## AGRADECIMENTOS

Gostaria de iniciar agradecendo meu orientador, Prof. Paulo Ricardo Lisboa de Almeida, por ter me acolhido e auxiliado durante todo o período de construção deste trabalho, desde a fomentação do tema até a sua execução. Gostaria também de deixar meus agradecimentos ao Prof. Luiz Eduardo Soares de Oliveira do Departamento de Informática da UFPR e ao Prof. Andre Gustavo Hochuli do PPGIA/PUCPR, que me auxiliaram durante o desenvolvimento deste projeto.

Além disso, gostaria de agradecer a minha família, que me acompanharam em cada etapa deste trabalho, escutando minhas angústias e comemorações, servindo de apoio para que eu pudesse finalizar este projeto, sem vocês eu provavelmente não teria conseguido alcançar meus objetivos. Especialmente, eu gostaria de agradecer meus pais, que se alegraram e me apoiaram continuamente, entendendo minhas ausências, correrias e me auxiliando sempre que possível.

Por último, mas não menos importante, eu gostaria de agradecer meus amigos seja eles da vida, faculdade ou trabalho. Vocês foram, provavelmente, as pessoas que mais escutaram sobre este trabalho, e o quanto ele estava me trazendo angústias e felicidades. Com vocês, fui capaz de aliviar o estresse e manter o foco, possibilitando que eu pudesse atingir meus objetivos com este projeto.

## RESUMO

Quando implementamos modelos de Aprendizado de Máquina em larga escala para suprir as necessidades de aplicações de cidades inteligentes, como o monitoramento de vagas de estacionamento por câmera, normalmente precisamos enviar os dados para grandes servidores para executarmos os processos de classificação. Isto pode ser particularmente complexo para a infraestrutura da cidade, pois os modelos de classificação de imagens requerem normalmente a transmissão de grandes volumes de dados, necessitando de uma estrutura de rede complexa e robusta. Buscando endereçar este problema nos cenários de classificação de vagas de estacionamento, este trabalho propõe uma arquitetura de destilação de modelos utilizando a técnica de Destilação de Conhecimento (*Knowledge Distillation* - KD), também conhecida como *Student-Teacher* (S-T). Neste processo, é utilizado um conjunto de modelos professores, que são destilados em modelos estudantes leves e especializados que podem ser executados diretamente em dispositivos de ponta. O processo de destilação de conhecimento e feito por meio de pseudo-rótulos geradas pelos professores, que são utilizadas para refinar os estudantes no cenário alvo. Os resultados demonstraram que os modelos estudantes, com 26x menos parâmetros que um modelo professor, alcançaram uma média de 96,6% de acurácia nos conjuntos de dados de teste, superando o resultado dos modelos professores que atingiram uma acurácia média de 95.3%, indicando que é possível realizar este processo de destilação sem perca de desempenho.



Palavras-chave: Cidades inteligentes. Aprendizado Profundo. Destilação de Modelos. Classificação de Vagas de Estacionamento. Computação em Dispositivos de Ponta.

## ABSTRACT

When deploying large-scale Machine Learning models for smart city applications, such as image-based parking lot monitoring, data often must be sent to a central server to perform classification tasks. This may be particularly challenging for the city's infrastructure, giving that image-based classification models normally require transmitting large volumes of data, necessitating complex and robust network infrastructure. Aiming to address this issue in image-based parking space classification, this work propose a model distillation pipeline, using the Knowledge Distillation (KD) technique, also know as Student-Teacher (S-T). In this process, one ensemble of teacher models is distilled in lightweight and specialized student models that can be deployed directly on edge devices. This knowledge distillation process happens with pseudo-labels generated by the teacher models, which are utilized to fine-tune the student models on the target scenario. The results demonstrate that the student models, with 26 times fewer parameters than one teacher model, achieved an average accuracy of 96.6% on the test datasets, surpassing the teacher models, which attained an average accuracy of 95.3%, demonstrating that it's possible to perform this distillation process whitout losing performance.



Keywords: Smart City. Deep Learning. Model Distillation. Parking Space Classification. Edge Computing.

## LISTA DE FIGURAS

## LISTA DE TABELAS

## LISTA DE ACRÔNIMOS

| | |
|---|---|
| KD | *Knowledge Distillation* |
| S-T | *Student-Teacher* |
| GPU | *Graphics Processing Unit* |
| TPU | *Tensor Processing Units* |
| JPEG | *Joint Photographic Experts Group* |
| MLP | *Multi-Layer Perceptron* |
| DNN | *Deep Neural Network* |
| CNN | *Convolutional Neural Network* |
| RGB | *Red, Green and Blue* |
| SVM | *Support Vector Machine* |
| CSV | *Comma Separated Values* |
| JSON | *JavaScript Object Notation* |

**SUMÁRIO**

# 1 INTRODUÇÃO

Modelos de Aprendizado de Máquina desenvolvidos para classificação de imagens normalmente necessitam de hardware especializado, como GPUs (*Graphics Processing Units*) ou TPUs (*Tensor Processor Units*), para lidar com a grande quantidade de dados de entrada. Por esse motivo, quando implementamos esses modelos em atividades reais de classificação, muitas vezes enfrentamos a necessidade de enviar os dados coletados para servidores que possuem poder computacional o suficiente para realizar estas atividades.

Em um contexto de cidades inteligentes, onde milhares de câmeras podem ser instaladas para alimentar os sistemas baseados em Aprendizado de Máquina, gargalos de dados podem surgir, tanto nos servidores quanto na infraestrutura de rede da cidade. De acordo com Satyanarayanan (2017), a transmissão de 12.000 *streams* de vídeo com resolução de $1920 \times 1080$ pode necessitar uma banda de 8,5 terabytes, gerando considerável pressão na estrutura de rede da cidade.

Para fornecer um melhor contexto do custo para o problema de classificação de vagas de estacionamento, considere uma instalação de baixa escala, contendo 1.000 câmeras e suponha que cada uma delas envia uma imagem JPEG de resolução $1280 \times 720$ comprimida para um servidor central a cada 30 segundos. Neste cenário, teríamos uma transmissão de 35 gigabytes de dados por hora para o servidor[1], não levando em consideração as sobrecargas adicionados pela rede. Este volume pode aproximadamente dobrar caso a resolução das imagens seja alterada para $1920 \times 1080$.

Este trabalho propõe a construção de uma arquitetura de destilação de modelos sob demanda, utilizando a técnica de *Student-Teacher* (S-T), para construir classificadores leves e especializados que podem ser executados diretamente em dispositivos de ponta, como câmeras inteligentes. A ideia principal é utilizar um modelo central robusto, capaz de classificar imagens de qualquer estacionamento, definido como professor, podendo ser computacionalmente caro e residindo no servidor. Desta forma, Toda vez que uma nova câmera é instalada em um estacionamento, as imagens são enviadas para este modelo por um curto período (por exemplo, 7 dias) para serem classificadas. Os resultados dessa classificação são utilizados como pseudo-rótulos para o treinamento de modelos menores e especialistas, definidos como estudantes, que podem ser instalados diretamente em câmeras inteligentes.

A hipótese de que modelos pequenos e especialistas podem ser usados em atividades reais é suportada por muitos trabalhos como (de Almeida et al., 2015; Amato et al., 2017; de Almeida et al., 2022; Ahrnbom et al., 2016), porém, seus autores utilizam dos rótulos reais existentes nos conjuntos de dados, criando problemas de escalabilidade, pois, desta forma, toda vez que um novo estacionamento é monitorado, seria necessário trabalho humano de rotulação das imagens para o treinamento dos classificadores.

Para suportar a hipótese de que modelos leves, refinados utilizando pseudo-rótulos, podem alcançar acurácia similar ao modelo professor, porém, com uma redução de poder computacional requerido, este trabalho visou responder as seguintes perguntas de pesquisa:

- Q1: Como é a acurácia do modelo estudante comparado ao modelo professor após refinamento?
- Q2: Quantas imagens o modelo professor necessita classificar para gerar pseudo-rótulos o suficiente para refinar o modelo estudante?

[1] Utilizando o conjunto de dados PKLot, podemos estimar que cada imagem JPEG compressada mantendo 95% de qualidade necessitaria de 303 kilobytes de dados.

- Q3: Como é a acurácia do modelo estudante refinado com pseudo-rótulos em comparação com um modelo estudante hipotético treinado com rótulos reais?

Na arquitetura, foram utilizados um conjunto de classificadores como modelos professores, onde cada modelo é uma Rede Neural Convolucional (*Convolutional Neural Network* - CNN) de estado da arte, denominada MobileNetV3 (Howard et al., 2019), composta de 4.204.594 parâmetros. Para os estudantes, foi aplicado um único modelo, sendo testadas duas arquiteturas compostas de 1.519.906 e 158.914 parâmetros. Um melhor detalhamento sobre as estruturas utilizadas nesse trabalho será fornecida no Seção 4.2.

Os resultados obtidos, detalhados no Capítulo 6, demonstraram que os modelos estudantes, refinados com os pseudo-rótulos e possuindo 26× menos parâmetros que um modelo professor, atingiram acurácia média acima de 96%, superando os resultados dos modelos professores, que atingiram valores médios de 95.3%, reforçando a hipótese de que modelos leves, refinados no estacionamento alvo, podem ser aplicados sem perca de desempenho. Desta forma, dentre as contribuições deste trabalho temos:

- Uma proposta de arquitetura de destilação de modelos, seguindo a técnica de *Student-Teacher*, para o problema de classificação de vagas de estacionamento.
- A publicação de um artigo internacional (Alves et al., 2024) na conferência *International Conference on Machine Learning and Applications* (ICMLA), 2024.

O restante deste documento está organizado da seguinte forma: O Capítulo 2 trata da fundamentação teórica sobre Aprendizado Profundo e Destilação de Conhecimento. O Capítulo 3 apresenta os trabalhos relacionados e de estado da arte sobre os temas envolvidos. No Capítulo 4 é apresentado o método proposto por este trabalho, explicitando as arquiteturas de CNNs utilizadas. O Capítulo 5 define o protocolo de treinamento das redes, além de apresentar os conjuntos de dados utilizados durante os experimentos. No Capítulo 6, os testes e resultados obtidos são detalhados. Por fim, no Capítulo 7, o trabalho é concluído, apresentado as respostas as questões de pesquisas elencadas anteriormente.

# 2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo introduz os conceitos básicos que fundamentam os assuntos discutidos neste trabalho. Inicialmente, será feita uma introdução aos temas de Redes Neurais, Aprendizado Profundo e Redes Neurais Convolucionais (*Convolutional Neural Networks* - CNNs), apresentando suas definições e objetivos. Em sequência, os temas de *Model lightweighting* e a técnica de Destilação de Conhecimento (*Knowledge Distillation* - KD), também conhecida como Estudante-Professor (*Student-Teacher* - S-T), serão abordados, introduzindo o leitor às ferramentas utilizadas na execução deste trabalho.

## 2.1 REDES NEURAIS

As Redes Neurais são um conjunto de técnicas de Aprendizado de Máquina que se baseiam no funcionamento do cérebro humano, por este motivo levam o nome de *Neural Networks*. Seu estudo iniciou-se em meados dos anos 50, tendo um grande marco com Frank Rosenblatt em 1957 com a implementação do *Perceptron* (Rosenblatt, 1958), sendo ele a unidade básica de uma Rede Neural, futuramente possibilitando a construção de estruturas mais complexas como *Multi-Layer Perceptrons* (MLP) e Redes Neurais Profundas.

### 2.1.1 *Multi-Layer Perceptron*

Um MLP é uma Rede Neural com mais de uma camada oculta (Bishop e Nasrabadi, 2006). Seu funcionamento baseia-se na transformação dos dados de entrada para uma dimensão onde o problema se torna linearmente separável, permitindo a sua divisão com funções lineares. Isto é proporcionado pelo encadeamento sucessivo de transformações não-lineares (Bishop, 1995) numa execução *feed-forward*, onde os dados trafegam em uma única direção, ou seja, de uma camada para a próxima até a execução da última.

O estudo e construção de redes com múltiplas camadas tornou-se viável a partir de 1986 com a popularização do algoritmo de propagação de erros conhecido como retro-propagação (Rumelhart et al., 1986). Até este momento, não haviam maneiras de relacionar os resultados de camadas intermediarias com o resultado da classificação, tornando complexo o treinamento destas redes. Com o algoritmo de retro-propagação, esta limitação foi removida, possibilitando a construção de algoritmos de treinamento, como o Descida de Gradiente (*Gradient Descent*), sendo ele e suas variantes amplamente utilizadas.

A estrutura de um MLP é dividida em camada de entrada (*input layer*), camadas escondidas (*hidden layers*) e camada de saída (*output layer*). Cada camada é constituída de neurônios, estruturas similares aos *Perceptrons*, porém, utilizando diferentes funções de ativação não lineares, como a *sigmoid*. O resultado da classificação é obtido pelo cálculo das suscetivas combinações lineares executadas por cada neurônio em adição da aplicação de suas funções de ativação. Um esquemático de um MLP de duas camadas pode ser encontrado na Figura 2.1, no qual $w_i$ são os pesos e $b_i$ os vieses (bias), sendo eles os parâmetros ajustáveis da rede, alterados durante o treinamento e utilizados no cálculo das combinações lineares aplicadas por cada neurônio. Como exemplificado na Figura 2.1, as saídas de cada camada são aplicadas como entrada na camada seguinte, sendo efetuado o mesmo processo até a determinação do resultado da classificação.

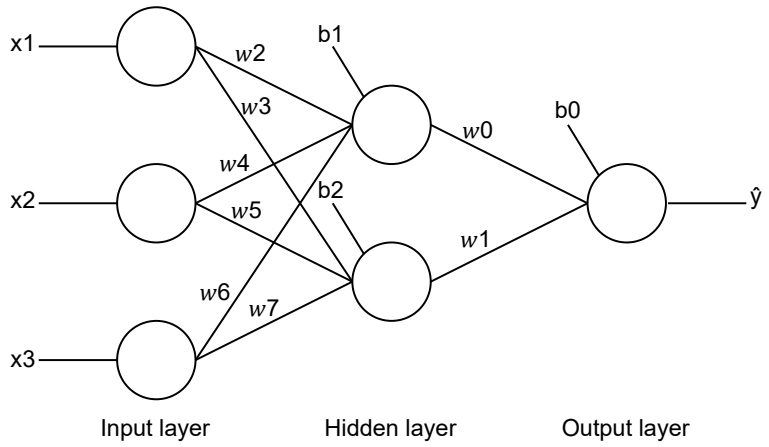



Figura 2.1: Exemplo de estrutura de um MLP.

Os MLPs, em conjunto com o *back-propagation*, proporcionaram novos caminhos nas pesquisas sobre Redes Neurais, possibilitando o surgimento de novas e mais complexas ferramentas. Em especial, as Redes Neurais Profundas (*Deep Neural Networks* - DNNs), estudadas amplamente na área de Aprendizado Profundo (*Deep Learning*).

## 2.2 REDES NEURAIS PROFUNDAS

Uma das grandes limitações na aplicação de Redes Neurais tradicionais reside em sua inabilidade de operar sobre dados puros. Durante décadas, a atividade de construção de extratores de características (*Feature Extractor*) que possuem a habilidade de transformar dados puros, como pixels de uma imagem, em representações computacionais que podem ser utilizadas por uma Rede Neural, exigiam alta dedicação e amplo conhecimento sobre o domínio do problema e Aprendizado de Máquina (LeCun et al., 2015).

Em contrapartida, o Aprendizado Profundo (*Deep Learning*) remove a necessidade de construir esses extratores de características, muitas vezes feitos por especialistas no domínio, construindo redes capazes de operar os dados em sua forma pura, cabendo a ela aprender como representar as informações, como descrito por LeCun et al. (2015):

> "Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics."

A construção de redes capazes de aprender a representação dos dados é alcançado com o encadeamento de múltiplas camadas escondidas (*hidden layers*), onde sua saída é relacionada com sua entrada seguindo uma representação hierárquica (Bishop e Bishop, 2023). Por exemplo, identificar a relação de pixels de uma imagem com a existência de um determinado objeto é algo relativamente complexo, porém, uma DNN pode aprender a identificar características (*features*) básicas, como bordas, em sua primeira camada, e utilizar estas características em suas camadas subsequentes compondo uma representação de mais alto nível, resultando no final na identificação do objeto (Bishop e Bishop, 2023).

De forma geral, podemos descrever as camadas de uma DNN como sucessivas transformações de dados que buscam representa-los em um novo espaço, onde eles possam ser separados em classes distintas. Desta forma, as últimas camadas de uma DNN podem ser vistas como um classificador linear, por exemplo, um MLP. O processo de encontrar uma representação através de uma transformação não-linear e a partir dos próprios dados é chamado de *representative learning* (Bishop e Bishop, 2023).

Com a adição de múltiplas camadas também aumentamos a quantidade de parâmetros ajustáveis da rede. Em comparação com os MLPs, que poderiam atingir a casa de centenas ou milhares de parâmetros, as DNNs podem alcançar a casa de centenas de milhares, milhões ou até mesmo bilhões de parâmetros. Logo, ao realizarmos a extração de características de forma automática com DDNs, aumentamos o custo e poder computacional necessário para utilização dessas redes.

O Aprendizado Profundo promoveu um grande crescimento na utilização de Redes Neurais, batendo recordes em atividades como reconhecimento de imagens e voz (LeCun et al., 2015). Dentre as possíveis arquiteturas de DNNs, temos as Redes Neurais Convolucionais, que são especializadas no tratamento de imagens e também foram os modelos utilizados neste trabalho.

### 2.2.1 Redes Neurais Convolucionais

As DNNs tradicionais proporcionaram vantagens consideráveis em relação aos MLPs, porém, em alguns casos, confeccionar tais redes torna-se computacionalmente inviável. Por exemplo, a construção de um classificador com 1.000 neurônios na primeira camada escondida e que recebesse como entrada uma imagem RGB $1000 \times 1000$, exigiria mais de 3 bilhões de parâmetros apenas nesta camada[1] (Bishop e Bishop, 2023). Entretanto, muitas aplicações trabalham com dados estruturados ou com boas correlações, ou seja, que possuem relações bem estabelecidas (Bishop e Bishop, 2023), de tal forma que é possível fazer uso delas para otimizar a construção de redes. Por exemplo, no caso de imagens, pode-se verificar que seus pixels são organizados em uma matriz bi-dimensional, onde os valores próximos possuem alta correlação.

Redes Neurais Convolucionais (*Convolutional Neural Networks* - CNNs), são uma arquitetura de redes que propõe mesclar a capacidade de uma DNN com a aplicação de matrizes de convolução, ou filtros, amplamente estudados na área de processamento de imagens (Gonzalez, 2009), buscando aproveitar a relação estrutural existente em imagens para construir redes mais eficazes para atividades como a detecção de objetos e classificação.

Em uma CNN, cada neurônio representa uma matriz de convolução (Bishop e Bishop, 2023). Desta forma, para uma matriz de tamanho $N \times N$, que atua sobre $C$ canais (dimensões) de entrada, temos um neurônio com $N^2C + 1$ ($N^2C$ pesos e 1 *bias*) valores ajustáveis, ou seja, parâmetros. Dessa forma, cada camada, composta de múltiplos neurônios, aplica múltiplos filtros aos valores de entrada, sendo elas denominadas camadas convolucionais, e a sua saída é chamada de *feature map*.

Em adição aos filtros de convolução, as CNNs aplicam técnicas de agrupamento (Bishop e Bishop, 2023), que possuem como objetivo reduzir o tamanho do *feature map*. Estas camadas são denominadas camadas de *pooling* e não possuem pesos ajustáveis, tendo seu funcionamento similar a aplicação de um filtro de convolução, porém, utilizando técnicas já pré-estabelecidas, como o *average-pooling* e o *max-pooling*, sendo o segundo o mais aplicado.

[1] O tamanho do vetor de entrada seria equivalente a $3 \times 1000 \times 1000 = 3.000.000$. Logo, desconsiderando os *bias*, cada neurônio da camada teria 3.000.000 parâmetros, totalizando 3.000.000.000.

Em suma, uma CNN é composta por uma sequência de camadas de convolução, seguidas ou não de *pooling*, que transformam a imagem de entrada em uma representação computacional válida, por meio da aplicação de diversos filtros de convolução (Bishop e Bishop, 2023), onde normalmente, após a última camada convolucional, é adicionado um MLP para a realização da inferência. Um diagrama de representação de uma CNN pode ser encontrado na Figura 2.2. Nela, os quadrados cinzas claros e escuros representam a aplicação de uma matriz de convolução ou técnica de agrupamento, exemplificando o processo de extração de características e construção do *feature map*, que posteriormente é enviado para o MLP para a realização do processo de classificação.

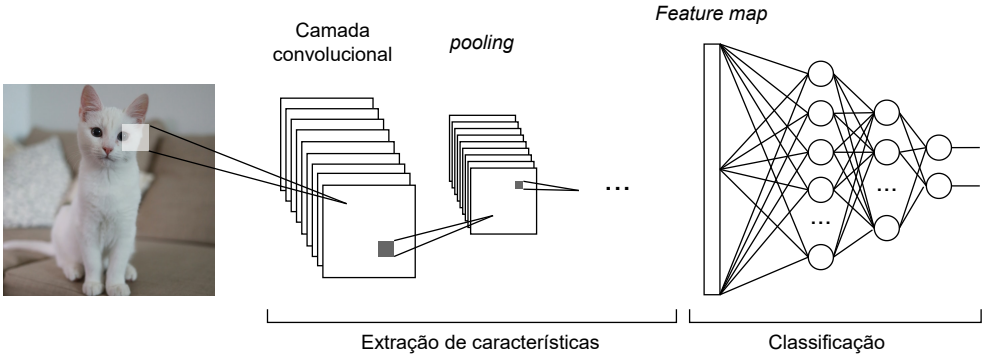



Figura 2.2: Exemplo de estrutura de uma CNN.

## 2.3 *MODEL LIGHTWEIGHTING*

Os modelos de Aprendizado Profundo têm se mostrado eficazes em diversas aplicações, sendo responsáveis por uma grande parcela das ferramentas baseadas em Redes Neurais utilizadas diariamente. Além disso, são detentores de grande parte dos recordes de desempenho em desafios de estado da arte, como, por exemplo, o *Model Soups* (Wortsman et al., 2022), alcançando uma acurácia *top-1* de 90,94% no desafio da *ImageNet*, que consiste de uma competição de classificação de imagens contendo 1.000 classes, no qual, a cada ano, os pesquisadores competem buscando atingir melhores resultados em sua classificação (Deng et al., 2009; Russakovsky et al., 2015). Entretanto, para alcançar estes resultados, estes modelos incorporam uma grande quantidade de camadas e, consequentemente, parâmetros, gerando alta demanda por poder computacional (Liu et al., 2024) e volume de dados categorizados (Wang e Yoon, 2021) para serem treinados e executados, também impossibilitando o seu uso em dispositivos com restrição de *hardware*, como, por exemplo, câmeras inteligentes.

Buscando endereçar este problema, o campo de estudo *Model Lightweighting*, também chamado de Compressão de Modelos (*Model Compression*), compõe um conjunto de técnicas de compressão, que visam reduzir o custo computacional necessário para a execução de DNNs. Dentre os conceitos desta área temos *Quantization* (Rokh et al., 2023), *Prunning* (Cheng et al., 2024), *Neural Architectural Search* (Ren et al., 2021) e *Knowledge Distillation* (Wang e Yoon, 2021), sendo este último o conceito básico utilizado neste trabalho.

### 2.3.1 Destilação de Conhecimento

A Destilação de Conhecimento (*Knowledge Distillation* - KD), engloba um conjunto de técnicas construídas com o objetivo de transferir conhecimento de um modelo, potencialmente

custoso, denominado professor, para outro modelo denominado estudante (Wang e Yoon, 2021), buscando reduzir tanto o poder computacional quanto a quantidade de dados categorizados necessários para o treinamento e utilização desses modelos. Por este motivo, o KD também pode ser referenciado como o *framework* de aprendizado Estudante-Professor (*Student-Teacher* - S-T) (Wang e Yoon, 2021). Em suma, as técnicas de KD podem ser aplicadas em dois campos de atuação distintos, sendo eles a Compressão de Modelos e a Transferência de Conhecimento.

Os algorítimos de KD podem ser divididos em 3 tipos denominados offline, online e auto-destilação, onde o critério de separação reside na definição do professor. Na destilação offline utilizamos um modelo professor já pré-treinado o qual não é modificado durante o treinamento do estudante, na online o professor pode continuar o seu treinamento junto ao estudante e na auto-destilação o modelo estudante torna-se seu próprio professor (Wang e Yoon, 2021).

Alguns trabalhos, como Wang e Yoon (2021), propõem subdivisões dos diversos algoritmos de KD, sendo o aspecto comum a transferência de conhecimento de um modelo para outro, seja qual for o meio escolhido. Neste trabalho, o método proposto reside na destilação offline, com enfoque no uso de pseudo-rótulos.

### 2.3.2 Destilação de Conhecimento com pseudo-rótulos

As técnicas de aprendizado supervisionado se baseiam na utilização de conjuntos de dados existentes na literatura. Estes conjuntos comumente são coletados e categorizados por um grupo de especialistas, e logo, possuem rótulos reais, também chamadas de *ground truth*. Pseudo-rótulos, por outro lado, são um conjunto de rótulos indicados por um não especialista, como, por exemplo, um modelo de Aprendizado de Máquina. Desta forma, os métodos de treinamento que os utilizam normalmente são categorizadas como aprendizado semi-supervisionado ou auto-supervisado (Wang e Yoon, 2021).

Em KD com pseudo-rótulos, a vasta maioria das técnicas utilizam o processo de aprendizado semi-supervisionado, construindo conjuntos de dados compostos tanto por rótulos reais quanto por pseudo-rótulos advindos do modelo professor (Wang e Yoon, 2021). Entretanto, neste trabalho, o método residiu na construção de conjuntos de dados compostos puramente por pseudo-rótulos, visando a remoção da necessidade de trabalho humano de rotulação, como exemplificado na Figura 2.3, que exemplifica o processo de classificação de um conjunto de dados não rotulados pelo modelo professor, gerando os pseudo-rótulos utilizados para o treinamento do estudante.

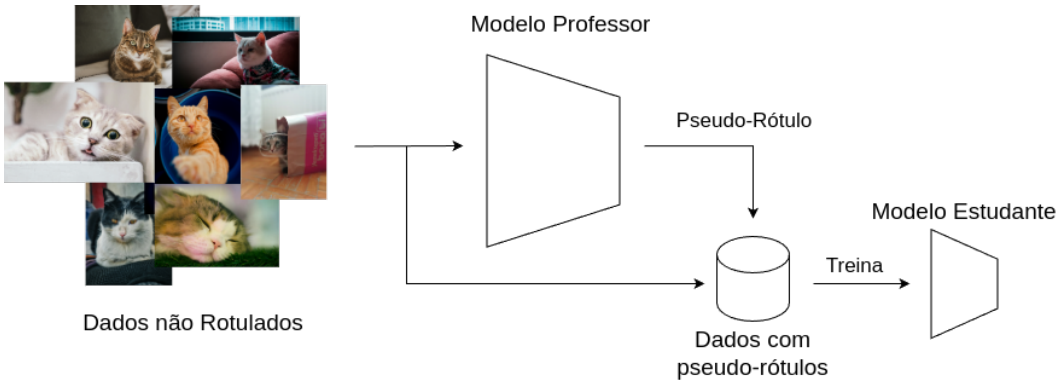



Figura 2.3: Exemplificação do fluxo de KD com pseudo-rótulos.

## 2.4 CONCLUSÃO

Neste capítulo foram abordados os conceitos básicos que nortearam a construção deste trabalho, apresentando as Redes Neurais, Aprendizado Profundo com a utilização de CNNs e as técnicas de Destilação de Conhecimento, descrevendo suas definições, objetivos e limitações. O próximo capítulo será destinado a apresentar o estado da arte e trabalhos relacionados a este tema, como a utilização de CNNs para classificação de vagas de estacionamento e aplicações de técnicas KD.

# 3 ESTADO DA ARTE

Aplicações de Aprendizado de Máquina para os problemas de classificação de vagas de estacionamento são amplamente utilizadas em trabalhos do estado da arte. Dentre os avanços notáveis, temos a criação de grandes conjuntos de dados de referência como a PKLot (de Almeida et al., 2015) e a CNRPark-EXT (Amato et al., 2017) que serão discutidos posteriormente. Em adição, o uso dos conceitos de Destilação de Conhecimento, com o modelo S-T, vem sendo extensamente abordados para produzir modelos leves (Wang e Yoon, 2021), e em diversos casos são utilizadas os pseudo-rótulos de maneira similar a este trabalho. Desta forma, este Capítulo tem o intuito de introduzir alguns dos recentes trabalhos em ambas estas áreas, apresentando as técnicas aplicadas e seus resultados.

## 3.1 CLASSIFICAÇÃO DE VAGAS DE ESTACIONAMENTO

Analisando o estado da arte e trabalhos recentes na classificação de vagas de estacionamento podemos separá-los em dois conjuntos, sendo eles os de **troca** e **não-troca**. Nos cenários de troca temos o treinamento de modelos gerais, onde os dados de treinamento são de uma câmera ou estacionamento diferentes dos dados de teste como em (de Almeida et al., 2015), (Amato et al., 2017), (Zhang et al., 2024), (Thakur et al., 2024), (Grbić e Koch, 2023), (Hochuli et al., 2023) e (Yuldashev et al., 2023). Já nos cenários de não-troca, os dados de treinamento e teste advêm da mesma câmera e estacionamento, explorados em (de Almeida et al., 2015), (Amato et al., 2017), (Grbić e Koch, 2023), (Zhang et al., 2024), (Yuldashev et al., 2023) e (Iqbal et al., 2021). Em adição, diversos trabalhos propõem o uso de classificadores leves, como em (de Almeida et al., 2015), (Amato et al., 2017), (Amato et al., 2018), (Zhang et al., 2024), (Hochuli et al., 2022), (Kolhar e Alameen, 2021) e (Nurullayev e Lee, 2019). Para uma análise mais detalhada veja de Almeida et al. (2022).

Por exemplo, em Amato et al. (2018) é proposto o uso de uma mAlexNet (Amato et al., 2016), que é baseada na AlexNet (Krizhevsky et al., 2012), composta de três camadas convolucionais e duas camadas densas, sendo implementada diretamente em uma câmera inteligente para a realização da classificação das vagas de estacionamento, não sendo necessário o envio de imagens para um servidor. Ademais, o método foi testado utilizando unicamente a CNRPark-EXT apresentando um *Overall Error Rate* de 0,4, sendo necessários 15 segundos para o processamento de uma imagem de entrada[1].

Mais recentemente, Hochuli et al. (2022) propôs uma CNN de 3 camadas convolucionais contendo aproximadamente 158.000 parâmetros, dimensionada especialmente para dispositivos com restrição computacional, como câmeras inteligentes. Entretanto, mesmo não exigindo imagens do conjunto de dados alvo para treinamento, este modelo atingiu apenas 80,9% de acurácia utilizando a PKLot e a CNRPark-EXT. Para comparação, a MobileNetV3 Large, com aproximadamente 4.1000.000 parâmetros atingiu uma acurácia de 89,9%.

De forma semelhante, Zhang et al. (2024) propõe um modelo leve para classificação de imagens de baixa resolução, buscando endereçar possíveis questões de segurança e privacidade pública relacionada a utilização de imagens de alta resolução. Entretanto, durante os testes de construção da rede, uma coleta aleatória de amostras da PKLot e da CNRPark-EXT foi efetuada para confecção dos conjuntos de treinamento e teste, ignorando o conceito temporal existente

[1] Não fica claro se os autores cronometraram o tempo para ambas classificação e sobrecargas, como o corte das imagens.

nestes conjuntos de dados e gerando possíveis vieses (de Almeida et al., 2022). Por fim, o método proposto atingiu em média 91,68% de acurácia utilizando a CNRPark-EXT, separando o conjunto em câmeras ímpares e pares, alternando entre treinamento e teste em um cenário de troca.

Já em Yuldashev et al. (2023) é proposta uma modificação da MobileNetV3 para a classificação de vagas de estacionamento. Esta versão é composta por diversas alterações de estruturas internas da rede, como módulos e funções de ativação, buscando especialização para o problema em questão. Ademais, os autores reportaram uma acurácia de 99% em cenários de não-troca e 95% a 98% em cenários de troca utilizando a PKLot e a CNRPark-EXT. Entretanto, a técnica de validação cruzada *k-fold*[2] foi aplicada durante o treinamento das redes, possivelmente gerando vieses nos resultados.

Em adição, Hochuli et al. (2023) estressa a comparação entre o uso de um único modelo ou diferentes técnicas de conjuntos de modelos para classificação em cenários de troca. No trabalho, é utilizado 3 diferentes arquiteturas de CNNs, variando seus tamanhos, e 4 diferentes conjuntos de dados. Os experimentos foram conduzidos utilizando a PKLot ou a CNRPark-EXT para treinamento e os demais conjuntos de dados para teste. Ademais, os autores reportaram que o uso de uma única MobileNetV3 atingiu melhores resultados em comparação com os outros modelos ou técnicas, com uma acurácia média de 95,5%. Entretanto, também constataram que a utilização de conjuntos de modelos pode trazer benefícios em situações onde os dados de treinamento são menos diversos, ou seja, com menos representatividade dos diversos contextos possíveis para estacionamentos.

Por outro viés, Grbić e Koch (2023) propõe a utilização de uma ResNet34 (He et al., 2015), um modelo pesado que, consequentemente, exige o uso de *hardware* especializado para sua aplicação. Desta forma, a rede foi testada utilizando a PKLot e a CNRPark-EXT nos cenários de troca e não-troca, reportando acurácias de 92% a 99% para o primeiro e acima de 99% para o segundo.

No mesmo contexto, Nurullayev e Lee (2019) propõe a CarNet, um modelo customizado para classificação de vagas de estacionamento que utiliza do conceito de camadas convolucionais dilatadas confeccionadas para aumentar o contexto de cobertura dos filtros de convolução. Ademais, os autores reportaram uma acurácia de 96% a 99% em cenários de não-troca e 94% a 98% em cenários de troca. Entretanto, de forma semelhante a Yuldashev et al. (2023), foram utilizados *k-folds* para o treinamento.

De forma semelhante, Thakur et al. (2024) propõe a utilização de duas redes pesadas distintas para a classificação de vagas de estacionamento, sendo elas a ResNet50 (He et al., 2015) e a VGG16 (Simonyan e Zisserman, 2015). Para a ResNet50, invés de alterar a camada de classificação, foi utilizado um SVM como classificador. Ademais, a PKLot foi utilizada para treinamento e o conjunto de dados *BarryStreet* (Acharya et al., 2018) para teste. Os resultados reportados são de 98,9% de acurácia para o primeiro modelo e 93,4% para o segundo. Entretanto, as informações relacionadas ao protocolo de divisão dos dados para o treinamento não deixam claro os critérios utilizados, logo, impossibilitando uma correta comparação de resultados.

Desta forma, como demonstrado na Tabela 3.1, mesmo com publicações reportando acurácias acima de 99%, a escalabilidade das soluções propostas ainda é uma questão em aberto já que, 1) Para atingir tais resultados, estas soluções se baseiam no uso de imagens rotuladas do conjunto de dados alvo, exigindo considerável esforço humano de rotulação; 2) Abordagens capazes de lidar com cenários de troca, onde as imagens de treinamento são de estacionamentos diferentes dos utilizados para teste, normalmente utilizam modelos computacionalmente caros, sendo necessário enviar estas imagens para um servidor para serem processadas. Mesmo assim,

[2]Consiste em separar os dados de treinamento em k grupos, utilizando, de forma alternada, $k-1$ grupos para treinamento e 1 para validação.

estas soluções raramente ultrapassam 95% de acurácia (de Almeida et al., 2022). Portanto, a definição e uso de classificadores mais acurados em cenários de troca ainda mostra-se uma questão evidente no estado da arte.

Tabela 3.1: Resumo dos trabalhos de classificação de vagas de estacionamento.

| Autores, Ano | Reproduzível | Acurácia (%) | | Conjuntos |
|---|---|---|---|---|
| | | Troca | Não-troca | |
| Amato et al. (2018) | Sim | - | 0,4[3] | CNRPark-EXT |
| Nurullayev e Lee (2019) | Sim | 94 - 98 | 96 - 99 | CNRPark-EXT PKLot |
| Iqbal et al. (2021) | Não | - | 97,6 | PKLot |
| Kolhar e Alameen (2021) | Não | 90 - 98 | 93 - 99 | CNRPark-EXT PKLot |
| Hochuli et al. (2022) | Sim | 80.9 | - | CNRPark-EXT PKLot |
| Yuldashev et al. (2023) | Sim | 95 - 98 | 99 | CNRPark-EXT PKLot |
| Hochuli et al. (2023) | Sim | 95,5 | - | CNRPark-EXT PKLot, Outros |
| Grbić e Koch (2023) | Sim | 92 - 99 | 99 | CNRPark-EXT PKLot |
| Thakur et al. (2024) | Não | 93,4 e 98,9 | - | BarryStreet PKLot |
| Zhang et al. (2024) | Sim | 91.7 | - | CNRPark-EXT PKLot |
| **Média** | - | 92,7 | 98,5 | - |

## 3.2 DESTILAÇÃO DE CONHECIMENTO COM PSEUDO-RÓTULOS

O uso de KD para construção de modelos leves tem grande abrangência em trabalhos do estado da arte, possuindo diversas formas de implementação (Wang e Yoon, 2021). Dentre as alternativas, a aplicação de pseudo-rótulos vêm sendo explorados para fazer uso dos dados não rotulados disponibilizados na internet. Dessa forma, nessa Seção são apresentados algumas das pesquisas utilizando KD com pseudo-rótulos, não seguindo necessariamente a mesma maneira de aplicação presente neste trabalho.

Em adição, Buciluǎ et al. (2006) é considerado um dos primeiros trabalhos a introduzir o conceito de compressão de modelos a partir da transferência de conhecimento de um classificador para outro. Neste trabalho, os autores buscaram endereçar o problema da complexidade e lentidão de grandes conjuntos de classificadores, os comprimindo em pequenas redes neurais a partir de dados pseudo-rotulados por estes classificadores. Além disso, os autores propuseram um novo algoritmo para confecção de pseudo-dados em cenários onde a existência de amostras

[3]Resultado informado em *Overall Error Rate* e não acurácia

não rotuladas não é muito evidente. Em seus resultados, demonstraram que as redes neurais, treinadas com um pequeno conjunto de dados reais em uma grande quantidade de pseudo-dados rotulados pelo conjunto, conseguiram atingir resultados virtualmente semelhantes ao conjunto de classificadores, de tal forma, introduzindo o conceito que posteriormente ficou conhecido como Destilação de Conhecimento.

Recentemente Ma et al. (2024) endereça o problema de detecção de ruas com o uso de KD para o aprendizado semi-supervisionado dos modelos, implementando uma ResNet101 e uma VGG16 como professores e a ResNet50 como estudante. Os professores foram treinados com uma combinação de aprendizado supervisionado em uma pequena quantidade de dados rotulados seguido por um processo de supervisão *cross-model* em uma grande quantidade de dados não rotulados. Por fim, as predições dos professores são utilizadas como pseudo-rótulos sendo aplicados em conjunto com uma pequena porção de dados rotulados para o treinamento do modelo estudante.

De forma semelhante, Manivannan (2023) faz uso do KD para o problema de detecção de falhas em superfícies. Em sua arquitetura, foram implementados um conjunto de ResNet18 como professores e uma única ResNet18 como estudante. Os professores foram refinados em partições distintas do conjunto de dados *TinyImageNet* seguido por um processo de treinamento colaborativo, utilizando os pseudo-rótulos geradas em imagens fracamente aumentadas como rótulos reais em imagens fortemente aumentadas. Por fim, o modelo estudante é treinado para minimizar o erro com dados rotulados e pseudo-rotulados advindos dos professores, enquanto busca equivaler as suas predições com as do conjunto professor.

De outra forma, Sun et al. (2021) propõe endereçar o problema de classificação de doenças raras por meio do uso de *Unsupervised Representation Learning* (URL) e auto-destilação com pseudo-rótulos. Neste trabalho, é aplicado uma ResNet12 como professor e estudante. Inicialmente, o modelo professor é treinado em um grande conjunto de dados base, não rotulado, de doenças comuns e controles normativos (*common diseases and normal controls* - CDNC), usando *contrastive learning*. Após este treinamento, o modelo é utilizado para gerar pseudo-rótulos de doenças raras a partir do mesmo conjunto de dados CDNC, sendo elas posteriormente utilizadas para o treinamento do modelo estudante.

Por outro viés, Xie et al. (2020) propõe um método iterativo para o treinamento de modelos através de aprendizagem semi-supervisionada e auto-treinamento. Nesta abordagem, os autores inicialmente treinam uma EfficientNet com dados rotulados, sendo ela posteriormente utilizada como professor. Em sequência, essa rede é aplicada para gerar pseudo-rótulos em imagens não rotuladas que são utilizadas em conjunto com dados rotulados para o treinamento do modelo estudante, possuindo tamanho equivalente ou maior que o professor. Este processo é repetido algumas vezes, onde, a cada iteração subsequente, o modelo estudante é aplicado como professor, rotulando as imagens novamente para o treinamento de uma nova rede seguindo o mesmo processo exemplificado anteriormente. Por fim, este método demonstrou resultados promissores nos desafios da *ImageNet*.

Já em Molo et al. (2024) o método proposto é o mais similar ao utilizado neste trabalho. Em sua arquitetura, é utilizado um grande modelo YOLO como professor, sendo executado em um servidor, e um pequeno, e leve modelo YOLOv5 como estudante. Os autores treinaram o modelo professor com uma combinação de conjuntos de dados do estado da arte para classificação de estacionamentos, como a PKLot. Apesar dos resultados promissores, o modelo professor foi destilado apenas na CNRPark-EXT e funciona mais como um detector do que um classificador e contém 1.760.518 parâmetros.

## 3.3 CONCLUSÃO

Neste Capítulo foram abordados alguns trabalhos relacionados a classificação de vagas de estacionamento e aplicação de técnicas de Destilação de Conhecimento, descrevendo brevemente seus processos de aplicação e resultados. Contudo, mesmo com os avanços existentes, a construção de modelos acurados capazes de serem instalados diretamente em dispositivos de ponta, sem a necessidade de esforço manual de rotulação de imagens, se mantêm uma questão em aberto no estado da arte. Desta forma, o próximo Capítulo será destinado em apresentar o método proposto neste trabalho para a destilação de modelos através de técnicas de KD, visando endereçar a questão mencionada acima.

# 4 MÉTODO PROPOSTO

Neste Capítulo será abordado o método proposto por este trabalho além das arquiteturas de redes utilizadas em sua fomentação. Inicialmente, a Seção 4.1 apresenta o esquema de funcionamento da arquitetura de destilação de modelos proposta, indicando como os professores e estudantes são construídos. Em sequência, a Seção 4.2 apresenta os modelos de CNNs aplicados neste trabalho.

## 4.1 ARQUITETURA DE DESTILAÇÃO DE MODELOS

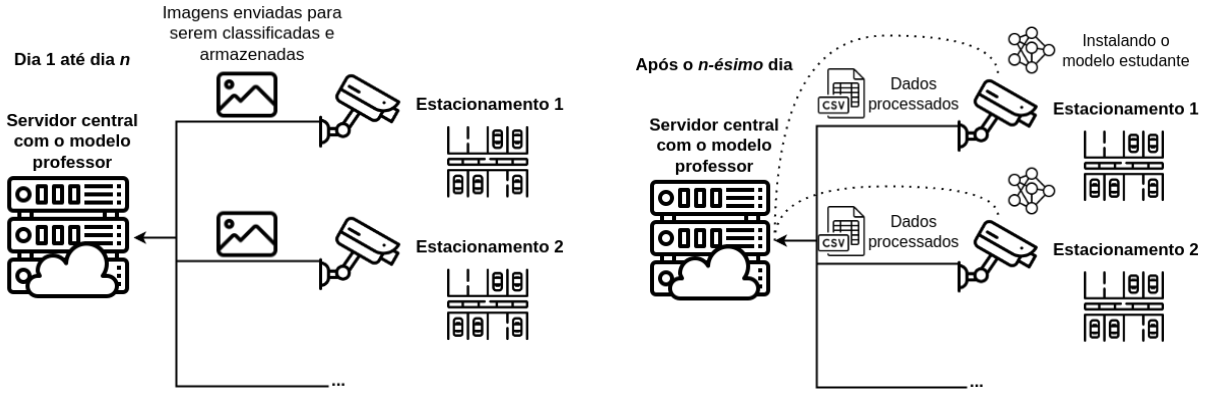
Diversos trabalhos demonstram que modelos de CNNs relativamente simples podem atingir resultados consideráveis quando fornecidas imagens do estacionamento alvo (de Almeida et al., 2015; Amato et al., 2017; Grbić e Koch, 2023; Amato et al., 2018; Nurullayev e Lee, 2019; Yuldashev et al., 2023). Inspirados nestes trabalhos, e seguindo o conceito de Destilação de Conhecimento, iniciado por (Buciluǎ et al., 2006) e também explorado por (Wang e Yoon, 2021; Molo et al., 2024), a seguinte arquitetura para destilação de modelos é proposta:

1. Um conjunto de classificadores é aplicado como modelo professor, buscando alcançar resultados equivalentes ao estado da arte em um cenário de *cross-dataset*, ou seja, onde nenhuma imagem do estacionamento de teste é apresentada durante o treinamento. Este conjunto pode ser computacionalmente caro e residir, por exemplo, num servidor. Mais detalhes sobre a construção deste conjunto serão apresentados no Capítulo 4.

2. A partir do momento que um novo estacionamento for adicionado a esta arquitetura, as suas imagens serão enviadas para o servidor por um período de $n$ dias (por exemplo, 7, como apresentados nos experimentos no Capítulo 6). Durante esta janela de tempo, o conjunto professor irá classificar as imagens, e aquelas classificadas com uma confiança *a posteriori* maior que 0,9 serão armazenadas como pseudo-rótulos.

3. Após o *n-ésimo* dia, o conjunto de pseudo-rótulos, obtido pelas classificações do modelo professor, será utilizado para realizar o treinamento do modelo estudante, sendo possível envia-lo diretamente para o dispositivo de ponta, por exemplo, uma câmera inteligente. A partir deste momento, não é mais necessário enviar as imagens cruas do estacionamento para o servidor. Em vez disso, apenas os dados já processados, como um arquivo *Comma Separated Values* (CSV) ou *JavaScript Object Notation* (JSON), contendo a localização das vagas e seus estados, precisa ser enviado para o servidor, visando fornecer as informações necessárias para os sistemas relacionados.

Uma ilustração desta arquitetura pode ser encontrada na Figura 4.1. Durante o período inicial de $n$ dias, como demonstrado na Figura 4.1a, as imagens de cada câmera instalada são enviadas para o servidor central, onde o conjunto professor as classifica, armazenando os pseudo-rótulos de alta confiança. Após os primeiros $n$ dias, o modelo estudante é refinado e enviado para a câmera. A partir deste momento, o dispositivo de ponta processa as imagens, enviando apenas as informações relevantes para o servidor central, como demonstrado na Figura 4.1b.




(a) Operação do dia 1 até dia $n$.

(b) Após o *n-ésimo* dia de operação.

Figura 4.1: Arquitetura proposta. Do dia 1 até o dia $n$ (a) as imagens são enviadas para serem classificadas pelo modelo professor e os resultados são armazenados. Após o *n-ésimo* dia (b), o modelo estudante é refinado com os pseudo-rótulos e enviado para a câmera.

## 4.2 CNN UTILIZADAS

Para o modelo professor foi utilizado a versão grande das redes MobileNetV3, denominada MobileNetV3 Large (Howard et al., 2019). Esta família de modelos foi projetada com o intuito de serem utilizadas em dispositivos moveis ou integrados, balanceando tamanho, velocidade de inferência e latência, enquanto mantendo uma acurácia razoável nos *benchmarks* do estado da arte. Desta forma, sua arquitetura implementa uma série de técnicas para melhorar os blocos convolucionais da rede, como conexões residuais, transformações não-lineares e blocos de *squeeze-and-excitation* que utilizam a relevância de cada canal da camada para computar o *feature map* de saída. Para mais detalhes desta arquitetura, veja Howard et al. (2019).

Como mencionado na Seção 4.1, o modelo professor compreende um conjunto (*ensemble*) de modelos, sendo cada um deles uma MobileNetV3 Large treinada separadamente (mais detalhes serão apresentados na Seção 5.2). Desta forma, considerando este modelo professor, e um dado de teste $x$, classe prevista $\hat{y}$ é dada pela média das probabilidades *a posteriori* da classe ser "ocupado"de todos os modelos do conjunto. A Equação 4.1 exemplifica o resultado obtido na classificação, sendo $T = [t_0, t_1, \ldots, t_i]$ o conjunto professor, composto de $k$ modelos e $P_i$ é a probabilidade *a posteriori* da classe ser "ocupado"dado pelo modelo $i$.

$$\hat{y} = \begin{cases} 1, & \text{se } f \geq \text{limiar}, \\ 0, & \text{caso contrário.} \end{cases} \qquad f = \frac{1}{|T|} \sum_{i \in T} P_i \tag{4.1}$$

Além disso, diferentemente do o modelo professor, os modelos estudantes são baseados em um único classificador, sendo testadas duas arquiteturas. A primeira consiste da versão pequena da MobileNetV3, denominada MobileNetV3 Small (Howard et al., 2019), utilizando das mesmas técnicas que a sua versão maior, porém, possuindo uma quantidade inferior de parâmetros. A segunda arquitetura constitui-se de uma CNN customizada e compacta, investigada em Hochuli et al. (2022, 2023), projetada para classificar vagas de estacionamento quando treinada com imagens do conjunto de dados alvo.

Sua arquitetura, detalhada na Figura 4.2, consiste em três camadas convolucionais combinadas com duas camadas de *pooling*, executando a classificação através de duas camadas densas e recebendo como entrada imagens em formato RGB de tamanho $32 \times 32$. A Tabela 4.1 compara as redes utilizadas neste trabalho, evidenciado as diferenças em quantidade de

parâmetros e tamanho ocupado em memória. Em especial, a Tabela 4.1 demonstra que o modelo customizado possui 26× menos parâmetros que a MobileNetV3 Large.

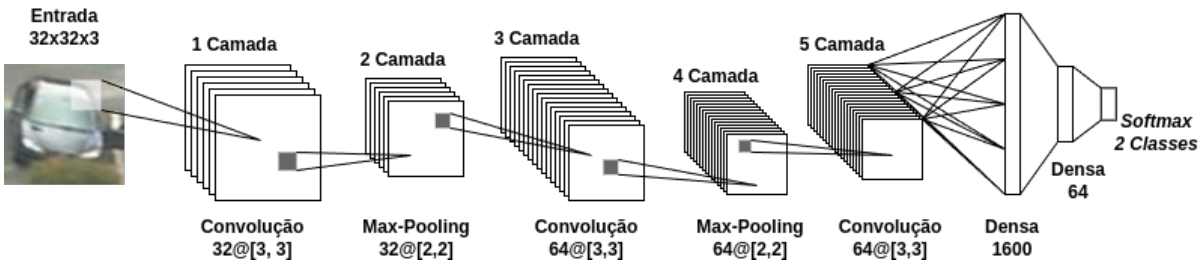



Figura 4.2: Modelo estudante customizado. consiste de 3 camadas convolucionais para realizar a extração de características e duas camadas densas para a classificação (Hochuli et al., 2022).

Tabela 4.1: Comparação das arquiteturas utilizadas neste trabalho.

| Arquitetura | # Parâmetros | Memória (IEEE 754 s.p.) |
|---|---|---|
| MobileNetV3 Large (Howard et al., 2019) | 4.204.594 | 17,00 MB |
| MobileNetV3 Small (Howard et al., 2019) | 1.519.906 | 6,20 MB |
| Customizada (Hochuli et al., 2022, 2023) | 158.914 | 0,64 MB |

## 4.3 CONCLUSÃO

Neste capítulo a arquitetura de destilação de modelos foi apresentada em detalhes, demostrando os passos executados por ela para atingir o objetivo de redução de custo computacional na classificação de vagas, sem a necessidade de esforço humano de rotulação de imagens de novos estacionamentos. Ademais, foram introduzidas a redes que propriamente executaram o fluxo em questão, pontuando suas principais propriedades. Desta forma, o próximo Capítulo focará na apresentação dos conjuntos de dados utilizados para validação desta proposta, além de apresentar o protocolo de treinamento dos modelos.

# 5 PROTOCOLO EXPERIMENTAL

Neste Capítulo serão apresentados os conjuntos de dados utilizados durante os experimentos deste trabalho, além de introduzir o protocolo de treinamento dos modelos professores e estudantes. Desta forma, a Seção 5.1 apresenta uma descrição completa dos conjuntos de dados, explicitando suas principais propriedades. Em sequência, a Seção 5.2 descreve o protocolo de treinamento utilizado na construção dos modelos aplicados durante os experimentos.

## 5.1 CONJUNTO DE DADOS

Nos experimentos executados nesse trabalho foram utilizados dois conjuntos de dados amplamente explorados no estado da arte, sendo eles a PKLot (de Almeida et al., 2015) e a CNRPark-EXT (Amato et al., 2017). A Tabela 5.1 apresenta as principais propriedades destes conjuntos de dados, demonstrando que a PKLot possui aproximadamente 8× mais imagens com três ângulos de câmera sobre dois estacionamentos, já a CNRPark-EXT possui uma quantidade inferior de amostras, porém, provê nove ângulos de câmera sobre o mesmo estacionamento. Além disso, ambos os conjuntos possuem um bom equilíbrio entre amostras de ambas as classes (ocupadas/vazias). Ao todo, 1.338.879 imagens de vagas de estacionamento foram utilizadas[1]. Todos os conjuntos possuem imagens retiradas de estacionamentos reais em diferentes épocas, ângulos de câmeras, estacões do ano e climas.

### 5.1.1 PKLot

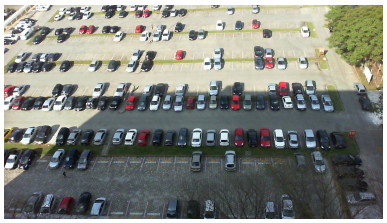
O conjunto de dados PKLot (de Almeida et al., 2015) contém imagens dos estacionamentos da Universidade Federal do Paraná (UFPR) e da Pontifícia Universidade Católica do Paraná (PUCPR). As imagens foram capturadas a cada cinco minutos por aproximadamente três meses sem intersecção entre os dias de coleta, apresentando três diferentes climas: ensolarado, nublado e chuvoso. As imagens foram armazenadas em resoluções de $1280 \times 720$ pixels em formato JPEG. Desta forma, foram coletadas 12.171 imagens, compondo 1.191.668 vagas de estacionamento distribuídas entre ocupadas e vazias.

Ao todo, dois estacionamentos foram abrangidos, possuindo dois ângulos de câmera no estacionamento da UFPR, denominados UFPR04 e UFPR05, posicionados respectivamente no 4° e 5° andar do prédio de administração da universidade e um ângulo de câmera no estacionamento da PUCPR, sendo posicionado no 10° andar do prédio de administração, e denominado PUCPR. Exemplos de imagem deste conjunto de dados pode ser encontrado na Figura 5.1.

### 5.1.2 CNRPark-EXT

O conjunto de dados CNRPark-EXT é uma extensão da CNRPark (Amato et al., 2017), contendo 4.073 imagens capturadas no campus do *National Research Concil* (CNR), em Pisa - Itália, propiciando 147.211 vagas de estacionamento. As imagens foram coletadas a cada 30 minutos de nove diferentes ângulos de câmeras simultaneamente e sobre o mesmo estacionamento, sendo armazenadas em resoluções de $1000 \times 750$ pixels. O conjunto engloba três climas: nublado, chuvoso, ensolarado. A CNRPark-EXT abrange uma grande quantidade de ângulos sobre o

[1] O numero total de imagens ultrapassa os valores originalmente publicados em (de Almeida et al., 2015; Amato et al., 2017), pois foram inseridas novas imagens rotuladas. Em adição, nos padronizamos as anotações e os retângulos rotacionados para todos os estacionamentos.



(a) Exemplo de imagem da câmera PUCPR.

(b) Exemplo de imagem da câmera UFPR04.

(c) Exemplo de imagem da câmera UFPR05.

Figura 5.1: Exemplos de imagens pertencentes ao conjunto de dados PKLot. As imagens (a), (b) e (c) representam dias ensolarados.

mesmo estacionamento, propiciando diferentes oclusões e desafios, dificilmente encontradas em outros conjuntos de dados. Exemplos pode ser encontrado na Figura 5.2.

(a) Exemplo de imagem da câmera 1.

(b) Exemplo de imagem da câmera 5.

(c) Exemplo de imagem da câmera 9.

Figura 5.2: Exemplos de imagens de algumas das câmeras pertencentes ao conjunto de dados CNRPark-EXT. As imagens (a), (b) e (c) representam dias ensolarados.

## 5.2 TREINAMENTO DAS REDES

Para o treinamento das redes foi utilizado a técnica de *cross-dataset*, ou seja, um conjunto de dados é aplicado para treinamento e validação do modelo e o outro para teste. Especificadamente, os experimentos foram conduzidos utilizando a PKLot para treinamento e a CNRPark-EXT para teste e vice-versa. Além disso, todos os subconjuntos produzidos durante os treinamentos foram nivelados a partir da classe menos recorrente.

Tabela 5.1: Conjuntos de dados utilizados nos experimentos.

| PKLot – 12.171 imagens | | | | | |
|---|---|---|---|---|---|
| # Dias | # Estacionamentos | # Ângulos | # Ocupado | # Vazio | # Total |
| 100 | 2 | 3 | 543.436 | 648.232 | 1.191.668 |
| CNRPark-EXT – 4.073 imagens | | | | | |
| # Dias | # Estacionamentos | # Ângulos | # Ocupado | # Vazio | # Total |
| 23 | 1 | 9 | 81.062 | 66.149 | 147.211 |

Em relação ao modelo professor, o seguinte método para construção do conjunto foi aplicado:

- Todos os classificadores professores utilizados foram pré-treinados no conjunto de dados *ImageNet*;
- Um primeiro classificador denominado $t_0$ foi treinado utilizando 70% dos dados de treinamento, referentes a cada ângulo de câmera e ordenados cronologicamente, sendo utilizados os 30% restantes para validação do modelo.
- Para cada ângulo de câmera $a_i$ disponível no conjunto de dados de treinamento, onde $i \in [1..k]$ é o índice de um possível ângulo de câmera $k$, foi criado um classificador $t_i$, treinado com todos os ângulos de câmera disponíveis, exceto $a_i$, sendo este utilizado para validação.

Dessa forma, o conjunto professor $T$ é formado pelos classificadores $T = [t_0, t_1, \ldots, t_k]$. Como discutido no Capítulo 4, as imagens dos primeiros $n$ dias do estacionamento de teste (alvo) são classificados pelo conjunto professor. Em sequência, os resultados da classificação são utilizados como pseudo-rótulos para o treinamento dos modelos estudantes.

Ademais, é criado um modelo estudante para cada ângulo de câmera. Cada estudante é inicialmente treinado usando o mesmo conjunto de dados utilizado no treinamento do modelo professor, e após isso é refinado utilizando as imagens e pseudo-rótulos referentes ao seu ângulo de câmera. Durante a fase de refinamento, foram utilizados os últimos $l = \lceil n/4 \rceil$ dias pseudo-rotulados como conjunto de validação, deixando os restantes $n - l$ dias para treinamento.

Para ambos os modelos professores e estudantes foi utilizado o otimizador *Adam* com uma taxa de aprendizado de 0,001 e mini-lotes de 64 imagens para treinamento. Durante o refinamento, apenas a última camada convolucional e as camadas densas das redes foram treinadas para os classificadores baseados na MobileNetV3. Por outro lado, todas as camadas foram treinadas para o modelo estudante customizado, por conta do seu menor tamanho. Todos os modelos foram treinados por 20 épocas e o classificador com melhor resultado nos dados de validação foi selecionado.

As imagens enviadas para as redes são em formato RGB, sendo o tamanho de $128 \times 128$ para os modelos professores e MobileNetV3 Small e $32 \times 32$ para o modelo estudante customizado. Todos os resultados reportados durante os experimentos são médias de 5 execuções isoladas.

## 5.3 CONCLUSÃO

Neste Capítulo foram introduzidos os conjuntos de dados utilizados durante os experimentos que serão detalhados no Capítulo 6. Além disso, o protocolo de treinamento dos modelos

professores e estudantes foi apresentado, explicitando o processo de construção do conjunto professor, além das técnicas utilizadas para a criação dos modelos estudantes. Desta forma, a próxima Seção será focada em introduzir os experimentos e resultados obtidos, visando responder as perguntas de pesquisa elencadas no início deste trabalho.

# 6 EXPERIMENTOS E RESULTADOS

Neste Capítulo os experimentos realizados neste trabalho serão detalhados. Desta forma, como comentado na Seção 5.2, foi adotado a técnica de *cross-dataset*, onde um conjunto de dados é utilizado para treinamento e o outro para teste. Portanto, a Seção 6.1, detalha os experimentos onde os modelos estudantes são refinados usando pseudo-rótulos advindos de sete dias de classificação pelo modelo professor. Na Seção 6.2, são apresentados os experimentos que demonstram a melhora nas redes conforme os dias pseudo-rotulados são aumentados. Por fim, na Seção 6.3, são detalhados os resultados considerando diferentes valores *a posteriori* para a classificação dos pseudo-rótulos.

## 6.1 UMA SEMANA DE PSEUDO-RÓTULOS

Nesta Seção, é apresentado os resultados dos testes onde, para os primeiros $n = 7$ dias, o modelo professor classifica as imagens do conjunto de dados alvo. De tal forma que, a partir do oitavo dia em diante, o modelo estudante refinado com os pseudo-rótulos gerados nos primeiros sete dias é utilizado para classificação.

Na Tabela 6.1, é sumarizado os resultados de acurácia nos conjuntos de dados de teste, excluindo os primeiros sete dias para evitar vieses. Ela demonstra a acurácia média para ambos os modelos customizado e MobileNetV3 Small (Modelos estudantes) após o refinamento usando os pseudo-rótulos. Também foram incluídos os resultados caso continuássemos a utilizar os modelos professores para classificar os dados de teste após o sétimo dia e as acurácias dos modelos estudantes antes do refinamento. A média ponderada apresentada na Tabela 6.1 considera ambos PKLot e CNRPark-EXT como conjuntos de teste, contabilizando o número de amostras em cada conjunto de dados.

Tabela 6.1: Resultados alcançados considerando $n = 7$ dias. Acurácia e ± desvio padrão.

| Treino PKLot - Teste CNRPark-EXT | | | | |
|---|---|---|---|---|
| | Rede Customizada | | MobileNetV3 Small | |
| Professor | Sem Refinamento | Refinado | Sem Refinamento | Refinado |
| $96,4\% \pm 0,2$ | $90,0\% \pm 1,0$ | $91,2\% \pm 1,0$ | $95,6\% \pm 0,6$ | $95,4\% \pm 0,3$ |
| Treino CNRPark-EXT - Teste PKLot | | | | |
| | Rede Customizada | | MobileNetV3 Small | |
| Professor | Sem Refinamento | Refinado | Sem Refinamento | Refinado |
| $95,2\% \pm 0,4$ | $80,2\% \pm 2,0$ | $97,2\% \pm 0,4$ | $91,3\% \pm 1,0$ | $97,2\% \pm 0,3$ |
| Média Ponderada | | | | |
| | Rede Customizada | | MobileNetV3 Small | |
| Professor | Sem Refinamento | Refinado | Sem Refinamento | Refinado |
| $95,3\% \pm 0,4$ | $81,1\% \pm 1,9$ | $96,6\% \pm 0,5$ | $91,7\% \pm 1,0$ | $97,0\% \pm 0,3$ |

Como demonstrado na Tabela 6.1, a MobileNetV3 Small refinada atingiu uma acurácia de 0,4 pontos percentuais a mais, em média, que a rede customizada. Entretanto, a MobileNetV3

Small tem aproximadamente uma ordem de magnitude a mais de parâmetros que o modelo customizado. Desta forma, este classificador pode balancear melhor a acurácia e eficiência computacional sob cenários com restrição de recursos.

É de conhecimento que o treinamento com pseudo-rótulos pode induzir vieses no modelo (por exemplo, qualquer erro cometido pelo modelo professor vai ser aprendido como um rótulo correto pelos estudantes). Porém, interessantemente, em média, os modelos estudantes alcançaram resultados melhores que os modelos professores, demonstrando que é possível economizar poder computacional, executando modelos leves em dispositivos de ponta, sem sacrificar acurácia. Isto é verdade mesmo para o modelo customizado, com apenas 158.914 parâmetros. Este fenômeno pode ser explicado por conta do pequeno número de parâmetros dos modelos estudantes, em adição ao pequeno número de pseudo-rótulos errados gerados pelo modelo professor (mais detalhes na Seção 6.3), o que pode ter ajudado a tornar os modelos estudantes especializados no cenário alvo.

Ao considerar o cenário onde os modelos estudantes foram refinados e testados unicamente na CNRPark-EXT, temos apenas um pequeno ganho na acurácia do modelo customizado e uma leva perda de acurácia para a MobileNetV3 Small. Entretanto, este resultado pode ser atribuído ao número limitado de amostras geradas durante o período de refinamento. No conjunto de dados CNRPark-EXT, as imagens foram capturadas em intervalos de 30 minutos, em um ângulo estreito, cobrindo apenas algumas vagas de estacionamento. Por exemplo, considerando a câmera 2 da CNRPark-EXT, em média, apenas 1.277 amostras pseudo-rotuladas foram geradas para refinar os modelos durante o período de sete dias examinado nesta Seção.

Por um motivo semelhante, durante os testes utilizando a PKLot, a acurácia foi melhorada significantemente após o refinamento (por exemplo, de 80,2% para 97,2% para o modelo customizado). Primeiramente, neste cenário, os modelos estudantes foram pré-treinados utilizando as imagens da CNRPark-EXT, que possui um número limitado de amostras e pode ter levado a uma má generalização dos modelos. Dando continuidade, com ângulos de câmera mais abertos e com imagens capturadas a cada cinco minutos, um maior número de pseudo-rótulos foi gerado na PKLot para refinar os modelos. Por exemplo, considerando a câmera UFPR04, 20.884 amostras pseudo-rotuladas foram geradas no mesmo período.

Por fim, a arquitetura foi testada em um Raspberry Pi 5, para verificar o tempo necessário para classificar vagas de estacionamento em dispositivos portáteis usando o modelo customizado. O sistema foi implementado utilizando Python, OpenCV e Pytorch, sem considerar nenhuma otimização. Para os testes, foi assumido que as imagens coletadas nas câmeras já estavam salvas na memória permanente do dispositivo. Desta forma, foi medido o tempo necessário para realizar o processo completo, desde carregar e recortar a imagem até a classificação de suas vagas.

Tabela 6.2: Tempo necessário para classificar vagas utilizando o modelo customizado.

| Conjunto | Vagas por Imagem | Tempo por Vaga (s) | Tempo por Imagem (s) |
|---|---|---|---|
| PKLot | 96 | 0,0104 | 1,0043 |
| CNRPark-EXT | 36 | 0,0097 | 0,3496 |
| Média | 66 | 0,01 | 0,6769 |

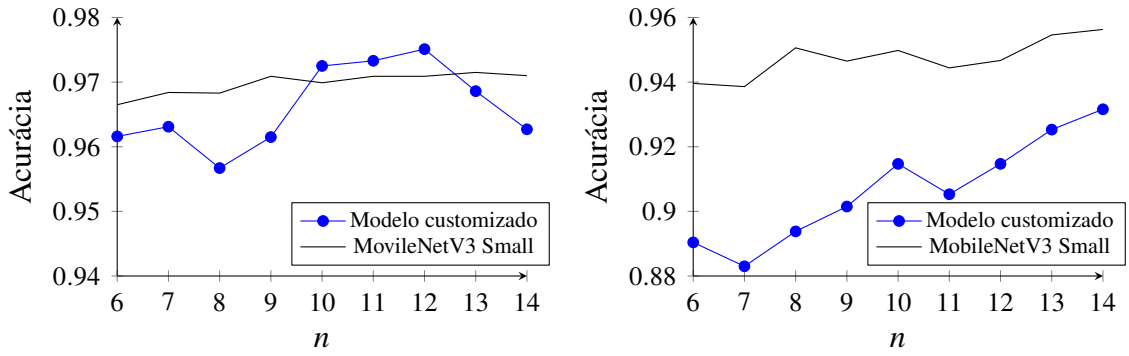
A Tabela 6.2 apresenta os tempos coletados neste experimento. Como demonstrado, em média, são necessários 0,01 segundos para processar cada vaga em ocupada/vazia. Portanto, caso seja considerado que uma câmera pode cobrir 100 vagas de estacionamento no seu campo de visão, será necessário 1 segundo para atualizar os estados de todas as vagas visíveis utilizando um Raspberry Pi 5. Porém, é importante apontar que este resultado não é diretamente comparável

com trabalhos do estado da arte, pois, foi utilizado uma versão mais recente dos dispositivos Raspberry Pi.

## 6.2 DIAS DE PSEUDO-RÓTULOS VERSUS ACURÁCIA

Nesta Seção, é examinado o desempenho dos modelos estudantes variando a quantidade de dias $n \in [6..14]$ para a coleta de imagens pseudo-rotuladas para o refinamento. Desta forma, buscamos analisar se o aumento de pseudo-rótulos para o refinamento melhora a acurácia dos modelos estudantes. Além disso, foram testados valores de $n \in [6..14]$, pois, seis dias era o valor mínimo necessário para gerar instancias de ambas as classes (ocupada/vazio) nos conjuntos de treinamento e validação, considerando todos os ângulos de câmeras disponíveis na PKLot e CNRPark-EXT. Ademais, como a CNRPark-EXT contém apenas 23 dias de dados coletados, ao considerarmos 14 dias de treinamento, nove dias foram deixados para testar os modelos.

Nos resultados apresentados, foram excluídos os primeiros 14 dias dos conjuntos de teste visando os deixar comparáveis, ou seja, todos os classificadores treinados, independente da quantidade de dias pseudo-rotulados, classificaram a mesma quantidade de imagens de teste. Desta forma, as Figuras 6.1(a) e 6.1(b) demonstram os valores obtidos considerando a PKLot e a CNRPark-EXT como conjuntos de testes, respectivamente. Como se pode observar, quanto maior o valor de $n$, melhor o resultado obtido, em média, em especial, considerando o modelo customizado. Estes resultados indicam que, quando possível, é interessante manter a arquitetura dependente do modelo professor por períodos maiores.




(a) PKLot como conjunto de teste.

(b) CNRPark-EXT como conjunto de teste.

Figura 6.1: Resultados variando o número de dias para refinar os modelos $n$. Em (a) a PKLot é utilizada como teste e em (b) a CNRPark-EXT é utilizada como teste.

Além disso, nas Figuras 6.1(a) e 6.1(b) é possível verificar momentos com quedas de acurácia, como em $n \in [12..14]$ para a PKLot como teste, e entre as quantidades $n$ de dias $6 \rightarrow 7$ e $10 \rightarrow 11$ para a CNRPark-EXT como teste. Este fenômeno pode ser explicado pela estrutura de coleta de imagens aplicado na construção dos conjuntos de dados. Em ambos, as imagens foram coletadas ao decorrer de semanas, abrangendo os finais de semana, nos quais a classe predominando são vagas vazias e, em alguns momentos, sem nenhuma amostra de vagas ocupadas. Desta forma, seguindo o protocolo de treinamento apresentado na Seção 5.2, pode ter ocorrido a construção de conjuntos de treinamento e validação pobres, decorrentes do nivelamento dos dados, produzindo modelos estudantes ineficientes.

Desta forma, a diversidade de imagens de ambas as classes (ocupadas/vazias) se mostra de grande importância para um resultado positivo, levantando a questão do cuidado ao selecionar os períodos de dias que as imagens seriam enviadas ao servidor para classificação, buscando evitar cenários com baixa diversificação (por exemplo, finais de semanas em ambientes mais ativos em

dias de trabalho), para obter bons conjuntos de treinamento e validação e, consequentemente, melhores modelos.

## 6.3 LIMIARES *A POSTERIORI* VERSUS ACURÁCIA

Nesta Seção, foi utilizado um procedimento similar ao usado na Seção 6.1, onde para os primeiros $n = 7$ dias, o modelo professor classifica as imagens e a partir do oitavo dia em diante, o modelo estudante é refinado com os pseudo-rótulos e utilizado para classificação. No entanto, nesta Seção foram variados os limiares *a posteriori* usados para selecionar os pseudo-rótulos. Desta forma, apenas as imagens classificadas com probabilidades *a posteriori* (dadas pelo modelo professor) maiores que $\in [0.5, 0.6, 0.7, 0.8, 0.9]$ foram selecionadas para refinar os modelos estudantes.

Neste cenário foi utilizado apenas o modelo customizado. Os resultados são apresentados na Tabela 6.3, no qual a acurácia se refere ao valor obtido após o refinamento do modelo estudante no conjunto de teste. A coluna *Usados* se refere ao total de amostras (somando todos os ângulos de câmera disponíveis em cada conjunto de teste) acima da probabilidade *a posteriori* mínima necessária, portanto, que foram utilizados como pseudo-rótulos. Já a coluna *Erradas* informa o total de imagens pseudo-rotuladas erroneamente pelo modelo professor que foram utilizadas no refinamento dos modelos estudantes. Além disso, a última linha da Tabela 6.3 apresenta o resultado de um oráculo hipotético, capaz de gerar pseudo-rótulos perfeitos para os conjuntos de testes, ou seja, os rótulos reais foram utilizados no treinamento. Este oráculo serve como um limite máximo para os resultados, demonstrando o melhor valor alcançável possível.

Tabela 6.3: Resultados alcançados considerando diferentes limiares *a posteriori* e os rótulos reais dos conjuntos de dados alvos. Resultados do modelo estudante customizado.

| | PKLot como conjunto de teste | | | CNRPark-EXT como conjunto de teste | | |
|---|---|---|---|---|---|---|
| *a p.* | Acurácia | Usadas | Erradas | Acurácia | Usadas | Erradas |
| 0,5 | $94,7\% \pm 0,7$ | 225.279 | 10.012 | $90,1\% \pm 0,7$ | 45.467 | 1.276 |
| 0,6 | $94,9\% \pm 0,7$ | 214.071 | 5.735 | $90,3\% \pm 0,9$ | 44.667 | 934 |
| 0,7 | $95,9\% \pm 0,6$ | 200.089 | 2.931 | $91,0\% \pm 0,6$ | 43.591 | 627 |
| 0,8 | $96,4\% \pm 0,6$ | 179.248 | 1.181 | $91,1\% \pm 0,6$ | 41.915 | 363 |
| 0,9 | $97,2\% \pm 0,4$ | 141.209 | 312 | $91,2\% \pm 1,0$ | 38.453 | 168 |
| Reais | $97,2\% \pm 0,4$ | 225.782 | - | $92,8\% \pm 0,6$ | 45.493 | - |

Os resultados apresentados na Tabela 6.3 mostram que, como esperado, quanto maior o limiar *a posteriori*, menos pseudo-rótulos errados são gerados para refinar os modelos estudantes, com o preço de diminuir a quantidade de amostras disponíveis para o treinamento destes modelos. Além disso, os resultados também demonstram que, ao utilizar pseudo-rótulos, é possível alcançar resultados similares aos se utilizar os rótulos reais. Desta forma, os valores apresentados aplicando um limiar *a posteriori* de 0,9 alcançaram virtualmente os mesmos valores que os modelos refinados com os rótulos reais ao considerar a PKLot como conjunto de teste, sendo apenas 1,6% abaixo dos modelos treinados com os rótulos reais no caso da CNRPark-EXT como conjunto de teste. Este comportamento pode ser explicado por conta do relativo baixo número de amostras rotuladas erroneamente pelo modelo professor quando consideramos o limiar de 0,9 (por exemplo, apenas 312, ou, 0,22% de 141.209 imagens foram rotuladas erroneamente na PKLot como conjunto de teste).

## 6.4 CONCLUSÃO

Neste Capítulo foram apresentados os experimentos realizados e resultados obtidos, buscando responder as questões de pesquisa elencadas no início deste trabalho. Os resultados se mostraram conclusivos, indicando que a arquitetura proposta é viável, possibilitando a construção de modelos estudantes leves e especializados nos estacionamentos alvos, além de viabilizar o seu uso em grande escala, como em cidades inteligentes. O próximo e último Capítulo será destinado para as conclusões finais deste trabalho, explicitando as respostas as perguntas de pesquisas inicialmente apresentadas.

# 7 CONCLUSÃO

Neste trabalho, foi proposto uma arquitetura de destilação de modelos, aplicando um conjunto de classificadores como modelo professor. Os professores são responsáveis por gerar pseudo-rótulos que em seguida são utilizados para o refinamento de modelos estudantes leves e especializados para a classificação de vagas de estacionamento. Desta forma, os estudantes foram refinados utilizando pseudo-rótulos específicos do ambiente em que seriam utilizados. Além disso, como modelos professores foram aplicadas as redes MobileNetV3 Large, já para os estudantes, um modelo customizado de três camadas e as MobileNetV3 Small foram utilizadas. Por fim, em relação as questões de pesquisa elencadas no início deste trabalho, as seguintes conclusões foram definidas:

**Q1: Como é a acurácia do modelo estudante comparado ao modelo professor após refinamento?** O modelo customizado de três camadas, que contém 26 vezes menos parâmetros do que as redes utilizadas no conjunto professor, conseguiu atingir resultados melhores do que os modelos professores após o refinamento utilizando os pseudo-rótulos, demonstrando que modelos leves, que podem ser instalados diretamente em dispositivos de ponta e com grande restrição computacional, podem atingir bons resultados sem a necessidade de rotulamento manual dos dados.

**Q2: Quantas imagens o modelo professor necessita classificar para gerar pseudo-rótulos o suficiente para refinar o modelo estudante?** Foi demonstrado que, em média, sete dias de imagens pseudo-rotuladas são o suficiente para refinar os modelos estudantes para atingir resultados melhores que o conjunto professor. Além disso, os experimentos também mostram que, quanto maior o tempo dedicado para o rotulamento de imagens pelo modelo professor, melhor o resultado alcançado pelos modelos estudantes. Porém, uma atenção deve ser dedicada ao selecionar o período no qual o modelo professor irá gerar os pseudo-rótulos, visando equilibrar a quantidade de amostras de ambas as classes, para construir conjuntos de treinamento e validação eficientes.

**Q3: Como é a acurácia do modelo estudante refinado com pseudo-rótulos em comparação com um modelo estudante hipotético treinado com rótulos reais?** Os experimentos executados demonstraram que, em média, apenas 0,27% dos pseudo-rótulos foram gerados erroneamente pelos modelos professores ao considerar um limiar *a posteriori* de 0,9. Portanto, os modelos estudantes refinados com os pseudo-rótulos atingiram uma acurácia de apenas 1,6 pontos percentuais a menos que os modelos refinados com os rótulos reais, quando consideramos a CNRPark-EXT, que foi o pior cenário estudado. Já para o caso da PKLot, os resultados foram virtualmente idênticos, demonstrando que é possível atingir valores equivalentes, em média, aos obtidos nos casos de treinamento com rótulos reais.

Por fim, os experimentos executados neste trabalho demonstraram que é possível utilizar modelos professores computacionalmente custosos para refinar modelos estudantes leves para o problema de classificação de vagas de estacionamento. Desta forma, torna-se viável a implantação destes sistemas de classificação em larga escala, como em cidades inteligentes, por exemplo. Em possíveis trabalhos futuros, é interessante a extensão do método aplicado para outros problemas de cidades inteligentes, como o monitoramento de tráfico e pedestres.

# REFERÊNCIAS


Acharya, D., Yan, W. e Khoshelham, K. (2018). Real-time image-based parking occupancy detection using deep learning. *Research@ Locate*, 4:33–40.

Ahrnbom, M., Astrom, K. e Nilsson, M. (2016). Fast classification of empty and occupied parking spaces using integral channel features. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Alves, P. L., Hochuli, A., de Oliveira, L. E. e de Almeida, P. L. (2024). Optimizing parking space classification: Distilling ensembles into lightweight classifiers. *arXiv preprint arXiv:2410.14705*.

Amato, G., Bolettieri, P., Moroni, D., Carrara, F., Ciampi, L., Pieri, G., Gennaro, C., Leone, G. R. e Vairo, C. (2018). A wireless smart camera network for parking monitoring. Em *2018 IEEE Globecom Workshops (GC Wkshps)*, páginas 1–6. IEEE.

Amato, G., Carrara, F., Falchi, F., Gennaro, C., Meghini, C. e Vairo, C. (2017). Deep learning for decentralized parking lot occupancy detection. *Expert Systems with Applications*, 72:327–334.

Amato, G., Carrara, F., Falchi, F., Gennaro, C. e Vairo, C. (2016). Car parking occupancy detection using smart camera networks and deep learning. Em *2016 IEEE Symposium on Computers and Communication (ISCC)*, páginas 1212–1217.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

Bishop, C. M. e Bishop, H. (2023). *Deep learning: Foundations and concepts*. Springer Nature.

Bishop, C. M. e Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

Buciluǎ, C., Caruana, R. e Niculescu-Mizil, A. (2006). Model compression. Em *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, páginas 535–541.

Cheng, H., Zhang, M. e Shi, J. Q. (2024). A survey on deep neural network pruning: Taxonomy, comparison, analysis, and recommendations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

de Almeida, P. R., Oliveira, L. S., Britto, A. S., Silva, E. J. e Koerich, A. L. (2015). Pklot – a robust dataset for parking lot classification. *Expert Systems with Applications*, 42(11):4937–4949.

de Almeida, P. R. L., Alves, J. H., Parpinelli, R. S. e Barddal, J. P. (2022). A systematic review on computer vision-based parking lot management applied on public datasets. *Expert Systems with Applications*, 198:116731.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. e Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. Em *CVPR09*.

Gonzalez, R. C. (2009). *Digital image processing*. Pearson education india.

Grbić, R. e Koch, B. (2023). Automatic vision-based parking slot detection and occupancy classification. *Expert Systems with Applications*, 225:120147.

He, K., Zhang, X., Ren, S. e Sun, J. (2015). Deep residual learning for image recognition.

Hochuli, A. G., Barddal, J. P., Palhano, G. C., Mendes, L. M. e Lisboa de Almeida, P. R. (2023). Deep single models vs. ensembles: Insights for a fast deployment of parking monitoring systems. Em *2023 International Conference on Machine Learning and Applications (ICMLA)*, páginas 1379–1384.

Hochuli, A. G., Britto, A. S., de Almeida, P. R. L., Alves, W. B. S. e Cagni, F. M. C. (2022). Evaluation of different annotation strategies for deployment of parking spaces classification systems. Em *2022 International Joint Conference on Neural Networks (IJCNN)*, páginas 1–8.

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V. e Adam, H. (2019). Searching for mobilenetv3. Em *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Iqbal, K., Abbas, S., Khan, M. A., Athar, A., Khan, M. S., Fatima, A. e Ahmad, G. (2021). Autonomous parking-lots detection with multi-sensor data fusion using machine deep learning techniques. *Computers, Materials & Continua*, 66(3).

Kolhar, M. e Alameen, A. (2021). Multi criteria decision making system for parking system. *Computer Systems Science & Engineering*, 36(1).

Krizhevsky, A., Sutskever, I. e Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.

LeCun, Y., Bengio, Y. e Hinton, G. (2015). Deep learning. *Nature*, 521(7553):436–444.

Liu, H.-I., Galindo, M., Xie, H., Wong, L.-K., Shuai, H.-H., Li, Y.-H. e Cheng, W.-H. (2024). Lightweight deep learning for resource-constrained environments: A survey. *ACM Computing Surveys*.

Ma, W., Karakuş, O. e Rosin, P. L. (2024). Knowledge distillation for road detection based on cross-model semi-supervised learning. Em *IGARSS 2024 - 2024 IEEE International Geoscience and Remote Sensing Symposium*, páginas 8173–8178.

Manivannan, S. (2023). Collaborative deep semi-supervised learning with knowledge distillation for surface defect classification. *Computers  Industrial Engineering*, 186:109766.

Molo, M. J., Carlini, E., Ciampi, L., Gennaro, C. e Vadicamo, L. (2024). Teacher-student models for ai vision at the edge: A car parking case study. *Proceedings Copyright*, 508:515.

Nurullayev, S. e Lee, S.-W. (2019). Generalized parking occupancy analysis based on dilated convolutional neural network. *Sensors*, 19(2):277.

Ren, P., Xiao, Y., Chang, X., Huang, P.-Y., Li, Z., Chen, X. e Wang, X. (2021). A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34.

Rokh, B., Azarpeyvand, A. e Khanteymoori, A. (2023). A comprehensive survey on model quantization for deep neural networks in image classification. *ACM Transactions on Intelligent Systems and Technology*, 14(6):1–50.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

Rumelhart, D. E., Hinton, G. E. e Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088):533–536.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C. e Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252.

Satyanarayanan, M. (2017). The emergence of edge computing. *Computer*, 50(1):30–39.

Simonyan, K. e Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.

Sun, J., Wei, D., Ma, K., Wang, L. e Zheng, Y. (2021). Unsupervised representation learning meets pseudo-label supervised self-distillation: A new approach to rare disease classification. Em *Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part V 24*, páginas 519–529. Springer.

Thakur, N., Bhattacharjee, E., Jain, R., Acharya, B. e Hu, Y.-C. (2024). Deep learning-based parking occupancy detection framework using resnet and vgg-16. *Multimedia Tools and Applications*, 83(1):1941–1964.

Wang, L. e Yoon, K.-J. (2021). Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE transactions on pattern analysis and machine intelligence*, 44(6):3048–3068.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S. e Schmidt, L. (2022). Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. Em Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G. e Sabato, S., editores, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 de *Proceedings of Machine Learning Research*, páginas 23965–23998. PMLR.

Xie, Q., Luong, M.-T., Hovy, E. e Le, Q. V. (2020). Self-training with noisy student improves imagenet classification. Em *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yuldashev, Y., Mukhiddinov, M., Abdusalomov, A. B., Nasimov, R. e Cho, J. (2023). Parking lot occupancy detection with improved mobilenetv3. *Sensors*, 23(17):7642.

Zhang, S., Chen, X. e Wang, Z. (2024). Bcfpl: binary classification convnet based fast parking space recognition with low resolution image. Em *International Conference on Image, Signal Processing, and Pattern Recognition (ISPP 2024)*, volume 13180, páginas 1442–1449. SPIE.