A Deep Learning Approach to Vegetation Images Recognition in Buildings: a Hyperparameter Tuning Case Study

André L. C. Ottoni and Marcela S. Novo

***Abstract*—Deep Learning methods have important applications in digital image processing. However, the literature lacks further studies that propose machine learning models to images classification in civil construction area. For example, the vegetation recognition on facades can be relevant in identifying the degradation and abandonment of buildings. Thus, the objective of this paper is to propose an Convolutional Neural Networks (CNN) approach to vegetation images recognition in buildings. For this, a database with urban images (low altitude) captured by a drone in Zurich (Switzerland) was adopted. In addition, a rigorous hyperparameters tuning methodology for the CNN model is presented. After adjusting the hyperparameters and the final model, the system achieved 90% of accuracy in the test stage. It should also be noted that CNN correctly classified 97.8% of the positive class (with vegetation on the facade) in test images.**

***Index Terms*—Deep Learning, convolutional neural networks, vegetation images recognition, hyperparameter tuning.**

I. Introdução

O aprendizado de Máquina, em inglês, *Machine Learning (ML)* é uma importante área da Inteligência Artificial (IA) [1], [2], [3]. Mais recentemente, os modelos de *Deep Learning* (Aprendizado Profundo) possibilitaram uma melhoria no processamento de imagens por técnicas de ML [4], [5], [6], [7]. Nesse sentido, na área da construção civil também podem ser destacados alguns exemplos de aplicação de algoritmos de ML no reconhecimento de imagens, como em [8], [9], [10]. No entanto, a literatura carece de mais estudos que proponham modelos de aprendizado de máquina para o processamento digital de imagens em tarefas específicas da Construção 4.0 [11], como por exemplo, no reconhecimento de imagens com vida vegetal em fachadas de edificações.

De fato, estudos sobre a análise de manifestações patológicas biológicas tem papel relevante na literatura [12], [13], [14], [15]. Isso porque, o crescimento de vegetações em construções é um problema que pode levar ao enfraquecimento das estruturas [12]. Além disso, o reconhecimento de vida vegetal em edificações pode ser relevante na identificação da degradação e abandono de edifícios históricos [12], [13], [16], [17]. Nesse aspecto, o trabalho de [13] investiga a presença de vegetação em edifícios históricos do Rio de Janeiro. Por outro lado, os trabalhos de [12] e [14] analisam a degradação de marquises (como por vegetação parasitária) em Recife (PE) e Campina Grande (PB), respectivamente. Nessa mesma linha, o estudo de [16] identifica que o crescimento de vegetação é uma das principais causas de deteorização de mesquitas Ottomanas.

A aplicação de modelos de *Deep Learning* para a detecção de vegetação em imagens também tem papel importante na literatura [18], [19], [20], [21], [22]. Os autores de [18] usam Redes Neurais Convolucionais (CNN) para classificar e segmentar imagens de satélite em cinco categorias: vegetação, solo, ruas, edifícios e corpos de água. O trabalho de [20], também utiliza CNN e sensoriamento remoto (dados lidar) para extrair informações de construções. Em outra via, o estudo realizado por [23] analisa mudanças em construções a partir da adoção da arquitetura de CNN, Inception_ResNet_V2. O processamento de imagens é dividido em etapas, sendo que na primeira fase (2D), as árvores foram detectadas e removidas para possibilitar a identificação de edifícios. Contudo, boa parte desses trabalhos concentra-se na análise de imagens obtidas por sensoriamento remoto (altas altitudes) [18], [20], [21]. Além disso, nesses trabalhos pouca atenção foi destinada para a influência do ajuste dos hiperparâmetros do desempenho de classificação da Rede Neural.

Realmente, um dos aspectos relevantes de sistemas de aprendizado é a definição das configurações iniciais da simulação (valores iniciais de taxas, algoritmos, arquitetura de uma rede neural) [24], [25]. Como destacado em diversos trabalhos, esses hiperparâmetros podem influenciar consideravelmente no resultado final do aprendizado [25], [26], [27]. Nesse aspecto, é frequente a análise da influência de hiperparâmetros de modelos de *Deep Learning* também em aplicações de processamento de imagens [28], [29], [30]. Por exemplo, vários trabalhos realizaram experimentos para verificar a influência de taxas de aprendizado em diferentes aplicações, como: tarefas diversas de segmentação de imagens [28], [30], detecção de fissuras em vidros [29], identificação de doenças em folhas de tomates [31] e classificação de imagens de raio-x [32]. Outras pesquisas investigaram a influência da definição dos otimizadores da CNN no processamento de imagens [29], [33], [30], [34], [35]. Nesse aspecto, em boa parte desses trabalhos, os autores destacam que a precisão de um modelo de CNN foi sensível ao ajuste desses hiperparâmetros de acordo com o conjunto de dados analisados [31], [33].

Dessa forma, o objetivo deste trabalho é aplicar Redes Neurais Convolucionais (CNN) para reconhecimento de imagens com vegetação em fachadas de edificações. Para isso,

André L. C. Ottoni, Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal da Bahia (UFBA), Salvador (BA) / Centro de Ciências Exatas e Tecnológicas da Universidade Federal do Recôncavo da Bahia (UFRB), Cruz das Almas (BA), Brasil, andre.ottoni@ufrb.edu.br

Marcela S. Novo, Departamento de Engenharia Elétrica e de Computação, Universidade Federal da Bahia (UFBA), Salvador (BA), Brasil, marcela.novo@ufba.br

foi adotada uma base de dados com imagens urbanas (baixa altitude) capturadas por um drone na cidade de Zurique na Suíça [36] para treinamento da arquitetura neural. Além disso, é apresentada uma criteriosa metodologia para seleção de hiperparâmetros de modelos de CNN. Nesse sentido, são investigados os efeitos dos otimizadores de atualização dos pesos, taxa de *dropout*, taxa de aprendizado e arquitetura da rede. Também é utilizada a técnica de geração de imagens (*data augmentation*) [5] para aumentar a variedade das imagens utilizadas para treinamento da CNN.

Este trabalho está estruturado em seções. A seção 2 apresenta conceitos sobre Redes Neurais. Na sequência, a seção 3 descreve as etapas da metodologia proposta. As seções 4 e 5, apresentam os resultados e um estudo comparativo com outros trabalhos, respectivamente. Finalmente, a seção 6 descreve as conclusões.

II. Redes Neurais Convolucionais

Uma Rede Neural Artificial (RNA) é um método de aprendizado de máquina supervisionado [1], [2], [3]. Nesse tipo de técnica, o treinamento dos modelos ocorre a partir de bases de dados rotuladas, ou seja, para determinados sinais de entrada sabe-se a saída desejada.

A representação original de uma RNA é inspirada na estrutura (dentritos, corpo celular e axônios) e função (conduzir impulsos elétricos) de um neurônio biológico [2]. Assim, um neurônio artificial é um modelo simplificado de um neurônio biológico [2], [3].

Os elementos básicos de uma RNA simples são [2]:

- Sinais de entrada: x_1, x_2, ..., x_n.
- Pesos sinápticos (w_1, w_2, ..., w_n.): são ajustados no processo de treinamento e ponderam as variáveis de entrada.
- Combinador linear: soma os sinais de entrada multiplicados pelos pesos.
- Potencial de ativação: valor produzido pelo combinador linear.
- Função de ativação: transforma o potencial de ativação em uma saída limitada. Alguns exemplos são as funções degrau, logística e tangente hiperbólica.
- Sinal de saída: y.

As Redes Neurais Convolucionais, em inglês, *Convolutional Neural Networks* (CNNs), são um tipo de RNA [37], [5]. As CNNs têm sido aplicadas com frequência na tarefa de processamento de imagens [4], [18], [5], [22], [10]. Isso se explica pela eficiência e robustez dessa técnica de *Deep Learning*, que trata as informações a partir de um grande número de camadas e neurônios. Além disso, diferentemente das RNAs tradicionais, as camadas das CNNs podem possuir distintas funções e tipos. Alguns exemplos são [5]:

- Camadas convolucionais: aplica a operação de convolução entre filtros e matriz de dados de entrada. Como saída, têm-se novas matrizes de acordo com o número de filtros convolucionais.
- *Pooling*: aplica transformações nos dados com o objetivo de diminuir as dimensões da matriz de entrada original. Por exemplo, são analisados os valores em um conjunto de 9 células (*kernel de pooling* 3×3) e apenas o maior valor (*max pooling*) é transferido para sequência das operações.
- *Flatten*: transforma as matrizes resultantes em uma única coluna de dados para ser utilizada como entrada de uma camada posterior.
- Camadas totalmente conectadas: semelhante às RNAs tradicionais ou *Multi Layer Perceptron* (MLP) , possuem um conjunto de neurônios e camadas totalmente conectadas.
- Camada classificadora: contêm neurônios que definem as classes de saída.

Outro método importante em RNAs e CNNs é o otimizador. O otimizador é responsável por ajustar os pesos de forma a mininimizar a diferença entre a saída desejada e a resposta da RNA [2]. A Equação 1 representa uma forma de atualização de pesos adotando o método de gradiente descente [2]:

$$w_{k+1} = w_k - T_A \frac{\partial E(w)}{\partial w} \tag{1}$$

em que, w_k são pesos e $E(w)$ é o erro quadrático produzido. Já T_A é a taxa de aprendizado, responsável por definir no otimizador a velocidade do processo de treinamento.

No processo de desenvolvimento de RNAs é comum a utilização de bibliotecas de funções, como por exemplo, o Keras [5]. O Keras reúne um conjunto de métodos que possibilitam o desenvolvimento rápido de modelos de Redes Neurais Convolucionais. Essa biblioteca é gratuita e disponível para a utilização nas linguagens Python ou R.

III. Metodologia

A. Base de Dados

Neste trabalho foram utilizados dados do *The Zurich Urban Micro Aerial Vehicle Dataset* (ZUDataset) [36] para treinamento e validação de Redes Neurais Convolucionais.

O ZUDataset é um conjunto de dados que contém 81.169 imagens gravadas por um vant das ruas urbanas de Zurique (Suíça). Um micro veículo aéreo (tipo quadrorrotor) registrou os dados em janeiro de 2015 com uma câmera GoPro Hero 4. As imagens foram registradas em alta resolução ($1920 \times 1080 \times 24$ bits). Além disso, as informações foram coletadas em baixas altitudes (5 a 15 m acima do solo) ao longo de percurso de 2 km de extensão.

O banco de dados também possui informações de GPS, medição inercial (IMU) e imagens do *Google Street View* ao nível do solo. Conforme informações no site do projeto (http://rpg.ifi.uzh.ch/zurichmavdataset.html), esse banco de dados é liberado sem restrições e pode ser usado para fins de pesquisa, avaliação e comerciais.

A Figura 1 apresenta exemplos de imagens capturadas pelo vant e disponíveis no ZUDataset.

De acordo com [36], o ZUDataset é ideal para aplicação em tarefas de navegação autonôma, como odometria visual, localização e mapeamento. No entanto, para a utilização neste trabalho, as imagens do ZUDataset foram selecionadas e adaptadas para a divisão em duas classes:

1) com vegetação na fachada da edificação;
2) sem vegetação na fachada da edificação.

(a)

(b)

Fig. 1. Exemplos de imagens do ZUDataset.

(a)

(b)

Fig. 2. Exemplos de imagens adaptadas do ZUDataset para a classe com vegetação.

No primeiro momento, foram analisadas as fotografias que continham vida vegetal nas paredes e realizado o procedimento de *zoom*, gerando novas imagens para um *dataset* adaptado para aplicação deste trabalho. Assim, foram obtidas 150 novas imagens para a classe 1 (com vegetação). A Figura 2 apresenta exemplos de imagens adaptadas do ZUDataset para a classe com vegetação na edificação.

(a)

(b)

Fig. 3. Exemplos de imagens adaptadas do ZUDataset para a classe sem vegetação.

Em seguida, foram selecionadas as fotografias das edificações que não continham vegetação nas fechadas das construções. Também foi aplicado o procedimento de *zoom* gerando 150 novas imagens para classe 2 (sem vegetação). A Figura 3 apresenta duas imagens adaptadas do ZUDataset para classe sem vegetação na construção.

B. Pré-Processamento

A base de dados (300 imagens), descrita na Seção 4.1, foi dividida aleatoriamente entre imagens para treinamento e validação do seguinte modo:

- 125 imagens da classe 1 (com vegetação) para treinamento.
- 125 imagens da classe 2 (sem vegetação) para treinamento.
- 25 imagens da classe 1 (com vegetação) para validação.
- 25 imagens da classe 2 (sem vegetação) para validação.

Os autores de [5] destacam a relevância da adoção de modelos de *Deep Learning* para conjunto de dados relativamente pequenos (*small data problems*). Para isso, podem ser realizadas algumas medidas para aumentar o desempenho do aprendizado profundo, como a aplicação de *data augmentation*. Nesse sentido, na etapa de pré-processamento das imagens foi utilizada dessa técnica de geração de imagens (*data augmentation*) [5]. Para isso, foram adotados os comandos *image_data_generator()* e *flow_images_from_directory()* da biblioteca Keras.

A função *image_data_generator()* gera lotes de dados com novas imagens modificadas a partir das originais. Esse processo torna-se relevante no trabalho com bases de dados

Fig. 4. Exemplos de imagens geradas pela função *image_data_generator()* do Keras.

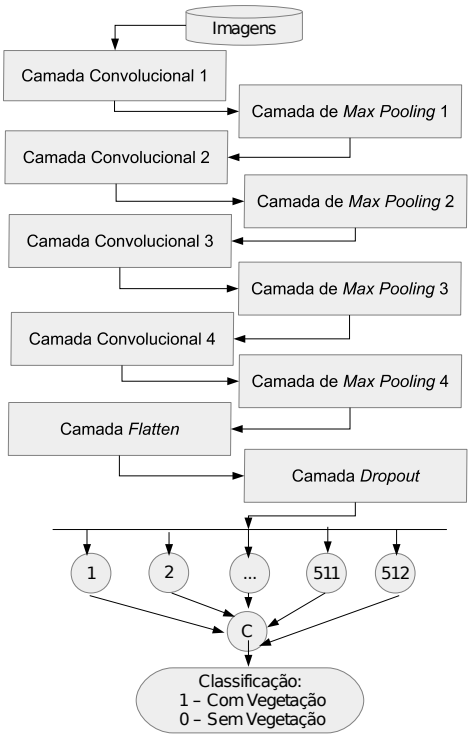


Fig. 5. Arquitetura da Rede Neural Convolucional base adotada (CNN-12). Baseada em [5].

originalmente pequenas [21], [32]. Nesse aspecto, foram utilizadas as seguintes transformações nas imagens originais para o aumento dos dados de treinamento: inserir rotação, aplicar ângulo de cisalhamento, usar *zoom*, virar as imagens horizontalmente e alterar as dimensões (altura e largura).

A Figura 4 exemplifica a aplicação das técnicas de aumento de dados pela função *image_data_generator()*. É possível perceber quatro imagens distintas geradas a partir de uma mesma fotografia original.

Também vale destacar a utilização da função *flow_images_from_directory()* do Keras. Esse método é responsável por realizar a leitura de lotes de dados em um diretório, com a adoção do gerador de imagens.

Quanto à normalização dos dados, foi utilizada a técnica de *min-max* [3], conforme Equação (2):

$$x_n = \frac{x - x_{min}}{x_{max} - x_{xin}}, \tag{2}$$

em que, x é um valor no conjunto de dados; x_{min} (0) é um valor mínimo; x_{max} (255) é um valor máximo; e x_n é o valor normalizado.

C. Arquitetura da Rede Neural

A arquitetura da Rede Neural Convolucional base (CNN-12) adotada é fundamentada em uma estrutura apresentada em [5]. A CNN-12 possui 12 camadas, sendo 4 convolucionais, 4 de *Max Pooling*, uma camada *Flatten*, uma de *Dropout*, uma camada densa com 512 neurônios e a camada de saída com o neurônio classificador (C). A Figura 5 apresenta a arquitetura utilizada como base nos experimentos.

Quanto aos hiperparâmetros da CNN-12, alguns foram fixados e outros selecionados para a realização de ajustes nas etapas experimentais. Os hiperparâmetros fixos durante as simulações foram: número de filtros em cada camada convolucional (32, 64, 128, 128), *kernel size* nas camadas convolucionais (3×3), *pool size* nas camadas de *Max Pooling* (2×2) e funções de ativação *relu* nas camadas convolucionais/densa e *sigmoid* da camada de classificação (C).

Por outro lado, os hiperparâmetros variáveis foram:

- Otimizador: *adadelta*, *adagrad*, *adam*, *rsmprop* e *sgd*.
- Taxa de *Dropout*: 0,2; 0,3; 0,4.
- Número de épocas de treinamento: 5; 10; 30.
- Taxa de aprendizado: 0,001; 0,005; 0,010.
- Uso do gerador de imagens: sim ou não.

Também foram realizados experimentos com mais quatro arquiteturas de RNA, baseadas em modificações na arquitetura base (CNN-12):

- MLP-6: arquitetura de 6 camadas e sem operações convolucionais. A estrutura possui uma camada de entrada *Flatten*, duas camadas densas de 512 neurônios uma de *Dropout*, mais uma camada densa com 512 neurônios e uma camada de saída com o neurônio classificador (C).
- CNN-8: arquitetura convolucional de 8 camadas. Nessa estrutura, foram retiradas duas camadas convolucionais (3 e 4) e duas camadas de *Max Pooling* (3 e 4) em relação arquitetura base CNN-12.
- CNN-10: arquitetura convolucional de 10 camadas. Nessa estrutura, foram retiradas uma camada convolucional (4) e outra de *Max Pooling* (4) em relação à arquitetura base CNN-12.

- CNN-14: arquitetura convolucional de 14 camadas. Nessa estrutura, foram adicionadas uma camada convolucional com 128 filtros e uma camada de *Max Pooling* (4) em relação à arquitetura base CNN-12.

A arquiteturas das RNAs foram codificadas a partir do comando *keras_model_sequential()*, disponível da biblioteca Keras da linguagem R [5].

D. Treinamento e Validação

Os processos de treinamento e validação são fundamentais para a utilização de uma RNA. No treinamento, os pesos da RNA são ajustados de acordo com um conjunto de dados apresentado. Para treinamento do modelo CNN foi utilizada a função *fit_generator()* do Keras. Enquanto que na fase de validação, é verificado qual a capacidade do modelo treinado classificar novas imagens.

Neste trabalho, é adotada como medida de desempenho principal da CNN o valor da acurácia (Ac) na etapa de validação, conforme Equação (3):

$$Ac = \frac{VP + VN}{VP + VN + FP + FN} \tag{3}$$

em que, VP é o número de verdadeiros positivos, VN são os verdadeiros negativos, FP é a quantidade de falsos positivos e FN são os falsos negativos.

A metodologia experimental proposta consiste em uma sequência de fases realizadas para viabilizar a seleção dos seguintes hiperparâmetros para a CNN: otimizador, taxa de *dropout* (T_D), taxa de aprendizado (T_A), adoção de gerador de imagens e arquitetura da rede neural. As etapas para treinamento e validação são descritas na sequência. Ressalta-se que nas etapas de 1 a 5 foi utilizada a arquitetura base CNN-12.

- Etapa 1: **Seleção de otimizadores**. Foram realizados experimentos com cinco tipos de otimizadores (*adadelta*, *adagrad*, *adam*, *rsmprop* e *sgd*). Para cada dos métodos foram executadas simulações de 3 repetições com 5 épocas de treinamento. Foram selecionados os dois otimizadores com a maior média de acurácia na validação. Nesta etapa, os demais hiperparâmetros foram mantidos como: $T_D = 0{,}3$, $T_A = 0{,}01$ e com gerador de imagens.
- Etapa 2: **Seleção da taxa de *dropout***. Foram realizados experimentos com três valores para T_D (0,2; 0,3; 0,4). Para cada valor de T_D foram executadas simulações de 3 repetições com 5 épocas de treinamento para cada um dos dois otimizadores selecionados na Etapa 1. Assim, foi selecionado um valor de *dropout* para cada um dos otimizadores. Nesta etapa, os demais hiperparâmetros foram mantidos como: $T_A = 0{,}01$ e com gerador de imagens.
- Etapa 3: **Seleção da combinação de otimizador e taxa de *dropout***. Foram realizados experimentos com duas combinações de otimizador e T_D definidos na Etapa 2. Para cada conjunto foram executadas simulações de 3 repetições com 10 épocas. Foi selecionada a combinação (otimizador e T_D) com a maior média de acurácia. Nesta etapa, os demais hiperparâmetros foram mantidos como: $T_A = 0{,}01$ e com gerador de imagens.
- Etapa 4: **Análise da influência do gerador de imagens**. Foram realizados experimentos com a adoção e sem a utilização de gerador de imagens (*data augmentation*) em 30 épocas. Foi adotado o método com a maior acurácia. Nesta etapa, os demais hiperparâmetros foram mantidos como: $T_A = 0{,}01$ e combinação de otimizador e taxa de *dropout* selecionados da Etapa 3.
- Etapa 5: **Seleção da taxa de aprendizado**. Foram realizados experimentos com três valores para T_A (0,01; 0,001 e 0,005). Para cada taxa de aprendizado foram executadas simulações de 3 repetições com 30 épocas. Foi selecionado o valor de T_A com a maior média de acurácia na validação. Nesta etapa, os demais hiperparâmetros foram mantidos como definidos nas Etapas 3 e 4.

Após a realização dos experimentos de treinamento e validação da etapa 5, os hiperparâmetros ajustados foram utilizados na fase de Teste, descrita na sequência.

E. Teste

Na etapa de teste, foram selecionadas 90 novas imagens de domínio público da internet. As fotografias de acesso livre e gratuitas foram obtidas no site Pixabay (www.pixabay.com), a partir de pesquisas de termos relacionados ao trabalho, como: "*vegetation building*", "vegetação construção", "edifício abandonado", "*building*" e "casa". Essas imagens continham, em geral, edificações abandonadas com vegetação na fachada (45 imagens de teste para a classe 1). Além disso, outras 45 imagens com construções sem vegetação para a compor a classe 2 no teste.

Os hiperparâmetros selecionados nas etapas anteriores foram utilizados para ajustar modelos de *Deep Learning* com cinco arquiteturas distintas adotadas neste trabalho: MLP-6, CNN-8, CNN-10, CNN-12 e CNN-14. Para cada estrutura neural foram realizadas 5 repetições com 30 épocas. Ou seja, o objetivo é verificar o desempenho dos hiperparâmetros selecionados e das arquiteturas de RNA, frente a um conjunto de dados totalmente novo.

Nesta fase, para avaliar a acurária foi utilizada o método *evaluate_generator()* do Keras [5]. Também foi adotada a função *predict_generator()* [5] para avaliar os resultados predição das classes (VP, VN, FP e FN).

IV. Resultados

Nesta seção, são apresentados os resultados referentes às fases de treinamento, validação e teste da aplicação de Redes Neurais Convolucionais. Em cada uma das etapas, foram selecionados os hiperparâmetros com as melhores métricas de acurácia para classificação de imagens com/sem vegetação na fachada de edificações.

A. Seleção de Otimizadores

A Tabela I apresenta os resultados de acurácia referentes à primeira etapa da metodologia de seleção de hiperparâmetros: análise de otimizadores.

TABELA I
RESULTADOS DE ACURÁCIA (%) PARA VALIDAÇÃO NA ETAPA 1 DE SELEÇÃO DE OTIMIZADORES.

Otimizador	Ac_1	Ac_2	Ac_3	Média
adadelta	50,0	49,8	80,3	60,0
adagrad	80,2	84,0	87,9	**84,0**
adam	49,8	49,8	50,1	49,9
rmsprop	49,8	50,0	50,2	50,0
sgd	76,4	81,2	77,6	**78,4**

TABELA II
RESULTADOS DE ACURÁCIA (%) PARA VALIDAÇÃO NA ETAPA 2 DE SELEÇÃO TAXA DE *dropout* (T_D).

Otimizador	T_D	Ac_1	Ac_2	Ac_3	Média
	0,2	78,0	81,6	82,1	80,6
adagrad	0,3	80,2	84,0	87,9	84,0
	0,4	88,3	82,0	81,8	**84,0**
	0,2	83,9	74,1	81,6	**79,9**
sgd	0,3	76,4	81,2	77,6	78,4
	0,4	77,8	76,1	78,2	77,4

A partir da Tabela I, nota-se que o desempenho do modelo CNN foi altamente dependente do método otimizador adotado. Além disso, é possível perceber que os dois otimizadores que alcançaram a maiores médias de acurácia na validação, foram: adagrad (84,0%) e sgd (78,4%). Assim, esses dois métodos foram selecionados para a sequência dos experimentos.

B. Seleção da Taxa de Dropout

Nesta etapa, o objetivo foi avaliar o desempenho dos otimizadores selecionados (etapa 1) com diferentes valores para taxa de *dropout* (T_D). A Tabela II ilustra os resultados dessa segunda fase.

Novamente, os experimentos revelaram uma importante influência dos hiperparâmetros nos resultados. Ao adotar o otimizador sgd, a taxa de *dropout* que gerou a maior média de acurácia foi $T_D = 0{,}2$ (79,9%). Por outro lado, dois valores de T_D (0,3 e 0,4) alcançaram os melhores resultados médios ao realizar os experimentos com o método adagrad. Assim, foi adotado como critério de desempate o maior resultado de acurácia em uma das repetições (88,3% em Ac_1). Dessa forma, foram selecionadas as seguintes taxas de *dropout* para a sequência dos experimentos: $T_D = 0{,}2$ (sgd) e $T_D = 0{,}4$ (adagrad).

C. Seleção da Combinação de Otimizador e Dropout

Na sequência, foram observados os desempenhos dos otimizadores e taxas de *dropout* em experimentos com mais épocas (10). A Tabela III apresenta os resultados referentes a terceira etapa de simulação.

É possível observar na Tabela III que a combinação (adagrad e $T_D = 0{,}4$) alcançou uma melhor média de acurácia (83,7%) para o treinamento em 10 épocas. Assim, após três etapas de treinamento e validação, foi selecionado o otimizador adagrad e taxa de *dropout* de 0,4 para as próximas fases.

TABELA III
RESULTADOS DE ACURÁCIA (%) PARA VALIDAÇÃO NA ETAPA 3 DE SELEÇÃO DA COMBINAÇÃO DE OTIMIZADOR E TAXA DE *dropout* (T_D).

Otimizador	T_D	Ac_1	Ac_2	Ac_3	Média
adagrad	0,4	86,2	88,0	77,0	**83,7**
sgd	0,2	84,0	82,0	82,3	82,8

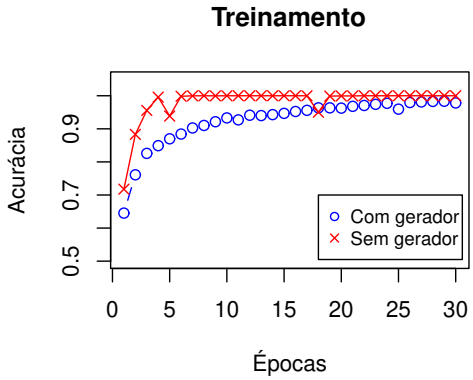


Fig. 6. Taxa de acurácia durante o treinamento para dois métodos: com gerador de imagens e sem gerador de imagens

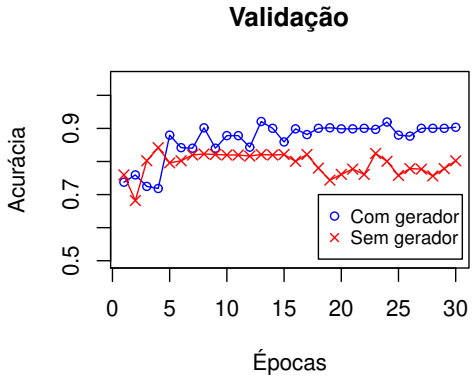


Fig. 7. Taxa de acurácia durante a validação para dois métodos: com gerador de imagens e sem gerador de imagens

D. Análise da Influência do Gerador de Imagens

Nesta etapa, foi realizada uma análise da influência da adoção do gerador de imagens (*data augmentation*). As Figuras 6 e 7 apresentam os resultados de acurácia nas etapas de treinamento e validação ao executar experimentos com: (i) gerador de imagens e (ii) sem gerador de imagens.

A partir da Figura 6, percebe-se que sem gerador de imagens, a CNN apresenta resultados de acurácia próximo de 1 (100%) em menos de 10 épocas de treinamento. No entanto, isso não se repete na etapa de validação (Figura 7 - acurácia em torno de 75%), quando são apresentadas para classificação um conjunto de imagens distinto da fase de treinamento. Isso é explicado, pois devido ao baixo número de imagens do banco

TABELA IV
RESULTADOS DE ACURÁCIA (%) PARA VALIDAÇÃO NA ETAPA 5 PARA SELEÇÃO DA TAXA DE APRENDIZADO (T_A).

T_A	Ac_1	Ac_2	Ac_3	Média
0,001	84,0	83,7	84,0	83,9
0,005	92,1	90,0	90,0	**90,7**
0,010	88,0	90,3	90,3	89,5

TABELA V
RESULTADOS DE ACURÁCIA (%) NA ETAPA DE TESTE DE ACORDO COM A ARQUITETURA DA REDE NEURAL.

Arq.	Ac_1	Ac_2	Ac_3	Ac_4	Ac_5	Média
MLP-6	81,1	78,9	78,9	75,6	76,7	78,2
CNN-8	84,4	90,0	87,8	90,0	82,2	**86,9**
CNN-10	78,9	77,8	80,0	81,1	90,0	81,6
CNN-12	87,8	78,9	86,7	88,9	90,0	86,4
CNN-14	84,4	81,1	86,7	75,6	85,6	82,7

de dados original, o modelo de CNN fica superajustado aos dados de treinamento, ou seja, uma situação de *overfitting* [2].

Por outro lado, nota-se na Figura 7 que ao adotar o método de geração de imagens, a Rede Neural Convolucional alcançou cerca de 90% de acurácia na validação. Assim, o método de geração de imagens possibilita incorporar ao banco de dados uma variedade maior de dados para treinamento, eliminando o processo *overfitting*. Nesse aspecto, conforme também demonstrado por [5], [32], ressalta-se a importância da utilização dessa técnica, principalmente para base de dados com um número pequeno de imagens em cada classe.

E. Seleção da Taxa de Aprendizado

Nesta etapa, são apresentados os resultados para experimentos com três valores para taxa de aprendizado: $0{,}001$, $0{,}005$ e $0{,}010$. Ressalta-se que nesta fase, as simulações foram realizadas em três repetições com 30 épocas em cada uma. A Tabela IV apresenta os resultados médios de acurácia para as diferentes taxas de aprendizado.

Conforme nota-se na Tabela IV, a taxa de aprendizado também influenciou no desempenho de classificação da rede neural. Nesse aspecto, a taxa de $0{,}005$ foi selecionada para a etapa seguinte, pois obteve a maiores médias de acurácia na validação ($90{,}7\%$).

F. Teste

Na etapa de teste, foram realizadas simulações com 5 arquiteturas de RNA: MLP-6, CNN-8, CNN-10, CNN-12 e CNN-14. Ressalta-se que no teste, foram adotadas imagens totalmente novas e distintas das fotografias utilizadas para o ajuste dos modelos. A Tabela V apresenta os resultados de acurácia nessa fase para cada uma das arquiteturas analisadas.

Em seguida, foi adotado a técnica de Análise de Variância (ANOVA) [38] para verificar se existe diferença estatística significativa entre os resultados de acurácia apresentados pelas cinco arquiteturas de RNAs. Assim, a ANOVA foi aplicada

TABELA VI
MATRIZ DE CONFUSÃO COM OS RESULTADOS PARA A ETAPA DE TESTE.

VP = 44	$FN = 1$
$FP = 8$	***VN = 37***

para avaliar se as médias populacionais (μ_i) são estatisticamente iguais ou diferentes, conforme hipóteses [38]:

$$\begin{cases} H_0: & \mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 \\ H_1: & \mu_i \neq \mu_j \quad \text{para pelo menos um par } i, j \end{cases}$$

Assim, a hipótese inicial (H_0) é aceita se existe igualdade de desempenho entre as cinco arquiteturas de RNAs analisadas ($p > 0{,}05$). Por outro lado, H_0 é rejeitada e a hipótese alternativa (H_1) é aceita se pelo menos um par de arquiteturas apresenta diferença significativa ($p < 0{,}05$), considerando os resultados de acurácia na fase de teste. O resultado da ANOVA indica que existe diferença entre as arquiteturas analisadas, sendo $p = 0{,}0145$ (critério de significância de 5%). Vale destacar que, as premissas de homogeneidade, normalidade dos resíduos e indepedência foram analisadas e respeitadas.

Após a confirmação que existe diferença estatística de desempenho ao adotar distintas arquiteturas de RNA no problema de classificação de imagens deste trabalho, foi selecionada a estrutura CNN-8. A arquitetura CNN-8 obteve a maior média de acurácia na etapa de teste ($86{,}9\%$). Além disso, o modelo selecionado (CNN-8) alcançou $90{,}0\%$ de acurácia como maior valor na etapa de teste em duas repetições (A_{C2} e A_{C4}). Esse valor equivale a classificação correta de 81 imagens, de um total de 90 fotografias da base de teste. A Tabela VI apresenta matriz de confusão para o teste, resultante da classificação do modelo ajustado com a adoção de CNN-8 (4ª repetição).

A partir da Tabela VI, é possível observar que a Rede Neural Convolucional selecionada (CNN-8) classificou corretamente 44 imagens da classe com vegetação ($VP = 44$), de um total de 45, o que equivale à $97{,}8\%$ de acertos na etapa de teste. Além disso, obteve acertos em 37 imagens (do total de 45) da classe sem vegetação ($VN = 37$). Por outro lado, destaca-se que a menor média de acurácia ($78{,}2\%$) foi através da arquitetura que não possui camadas convolucionais (MLP-6). Esses resultados indicam a importância da utilização de uma arquitetura convolucional para a tarefa de classificação de imagens analisada. Além disso, a relevância da seleção dos hiperparâmetros criteriosamente definidos nas etapas de treinamento, validação e teste. Os hiperparâmetros do modelo final são apresentados na Tabela VII.

V. COMPARAÇÃO COM OUTROS TRABALHOS

Nesta seção, a Tabela VIII apresenta uma comparação da presente proposta com outros trabalhos que aplicaram *Deep Learning* no reconhecimento de vegetação em imagens urbanas: I [18], II [21] e III [22].

A Tabela VIII revela que os estudos analisados dedicam-se principalmente à tarefa de segmentação de imagens urbanas, incluindo vegetação. Em outra via, a proposta deste trabalho é voltada para o processo de classificação. Nesse sentido, uma

TABELA VII
HIPERPARÂMETROS SELECIONADOS.

Hiperparâmetro	Selecionado
Otimizador	adagrad
Taxa de *dropout*	0,4
Gerador de imagens	Sim
Taxa de aprendizado	0,005
Número de épocas	30
Arquitetura	CNN-8

TABELA VIII
COMPARAÇÃO DA PRESENTE PROPOSTA (PROP.) COM DIFERENTES ESTUDOS QUE APLICARAM DEEP LEARNING NO RECONHECIMENTO DE VEGETAÇÃO EM IMAGENS URBANAS: I [18], II [21] E III [22].

		Prop.	I	II	III
Aplicação de	Classificação	✓	✓	–	–
CNN	Segmentação	–	✓	✓	✓
Tipo de	Fachadas de Construções	✓	–	–	–
dataset	Imagens de Satélite	–	✓	✓	–
	Fotografias Áereas	✓	–	–	✓
Hiperparâmetros	Otimizador	✓	–	–	–
ajustados	*Dropout*	✓	–	–	–
	Data Augmentation	✓	–	✓	–
	T_A	✓	–	–	–
	Arquitetura	✓	✓	✓	✓
Acurácia Máx.	$\geq 90\%$	✓	✓	✓	✓

importante contribuição do presente estudo é a aplicação de *Deep Learning* no reconhecimento de vegetação em fachadas de construções, a partir de imagens áreas de baixa altitude ou fotografadas em solo. Na literatura, boa parte dos estudos é realizado com imagens retiradas em alta altitude, como a partir de satélites.

Também vale destacar que a presente proposta adota uma metodologia para selecionar hiperparâmetros relevantes nas etapas de treinamento e validação de um modelo CNN: otimizador, *dropout*, *data augmentation* e taxa de aprendizado. Os demais trabalhos analisados na Tabela VIII, focaram principalmente no ajuste de características da arquitetura do modelo neural, como: número de camadas e número de filtros.

Por fim, destaca-se que o modelo CNN selecionado alcançou acurácia de 90% na etapa de teste, sendo que $97{,}8\%$ na classificação de verdadeiros positivos. Esse patamar de taxa de acertos ($\geq 90\%$) também foi aprensentado nos resultados dos estudos analisados [18], [21], [22], indicando um bom ajuste do modelo proposto na tarefa de reconhecimento de vegetação em edificações.

VI. CONCLUSÃO

O presente trabalho apresenta como principais contribuições e resultados: (i) aplicação de modelos de *Deep Learning* no processo de reconhecimento de vegetação em fachadas de edificações; (ii) proposta de uma criteriosa seleção de hiperparâmetros de modelos CNN adotando conceitos de estatística; (iii) adoção da técnica de ANOVA para analisar a diferença estatística entre cinco arquiteturas neurais na etapa de teste; (iv) utilização da técnica de geração de imagens (*data augmentation*) para aumentar a variedade de imagens no *dataset* de treinamento; (v) acurácia nas etapas de treinamento, validação e teste maiores ou iguais a 90% após o ajuste dos hiperparâmetros.

Os resultados demonstraram que todos os hiperparâmetros analisados influenciaram diretamente no desempenho da CNN na classificação binária de imagens em: (i) com vegetação na fachada e (ii) sem vegetação na fachada. Por exemplo, a acurácia média dos cinco otimizadores analisados (etapa 1) variou entre $49{,}9\%$ e $84{,}0\%$. Também vale destacar que a adoção do gerador de imagens possibilitou a eliminação de *overfitting*, aumentando a acurácia nos dados de validação. Além disso, o método proporcionou uma taxa de acertos de até $92{,}1\%$ na etapa de validação.

Também vale destacar os experimentos realizados com cinco arquiteturas de Redes Neurais Artificiais. A partir da técnica de ANOVA foi possível observar que existe diferença estatística significativa ao adotar distintas estruturas de *Deep Learning* na tarefa de classificação de imagens deste trabalho. A arquitetura com 8 camadas (CNN-8) foi selecionada, pois alcançou a maior média de acurácia na etapa de teste ($86{,}9\%$).

Após a seleção dos hiperparâmetros e ajuste de diferentes arquiteturas de RNAs, o sistema alcançou $90{,}0\%$ de acurácia na etapa de teste. Ressalta-se ainda que a CNN classificou corretamente $97{,}8\%$ das imagens de teste da classe positiva: com vegetação na fachada.

Em trabalhos futuros, espera-se aplicar Redes Neurais Convolucionais para classificar outras características possivelmente encontradas em fachadas de edificações [39]. Além disso, espera-se expandir a adoção de métodos estatísticos de inferência e planejamento de experimentos [38], [27] para aprimorar a metodologia de análise e seleção de hiperparâmetros da CNN.

AGRADECIMENTOS

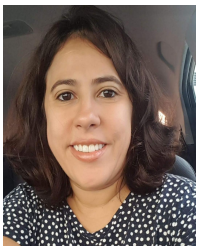
Agradecemos ao PPGEE/UFBA e UFRB. Também agradecemos aos pesquisadores da Universidade de Zurique por disponibilizarem a base de dados (treinamento e validação) em domínio público e aos autores da fotografias disponibilizadas no Pixabay (adotadas na fase de teste).

REFERÊNCIAS

[1] S. J. Russell and P. Norving, *Artificial Intelligence*. Campus, 3st ed., 2013.
[2] I. N. Silva, D. H. Spatti, and R. A. Flauzino, *Redes Neurais Artificiais Para Engenharia e Ciências Aplicadas: fundamentos teóricos e aspectos práticos*, 2nd ed., Ed. ArtLiber, 2016.
[3] L. A. da Silva, S. M. Peres, and C. Boscarioli, *Introdução à mineração de dados: com aplicações em R*. Elsevier Brasil, 2017.
[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
[5] F. Chollet and J. J. Allaire, *Deep Learning With R*. Manning Publications, 2018.
[6] L. G. C. Evangelista and E. B. Guedes, "Ensembles of convolutional neural networks on computer-aided pulmonary tuberculosis detection," *IEEE Latin America Transactions*, vol. 17, no. 12, pp. 1954–1963, 2019.
[7] O. L. de Sousa, D. M. Magalhães, P. d. A. Vieira, and R. Silva, "Deep learning in image analysis for covid-19 diagnosis: a survey," *IEEE Latin America Transactions*, vol. 100, no. 1e, 2020.

[8] F. Pirotti, C. Zanchetta, M. Previtali, and S. Della Torre, "Detection of building roofs and facades from aerial laser scanning data using deep learning," in *2nd International Conference of Geomatics and Restoration, GEORES 2019*, vol. 42, no. 2. Copernicus GmbH, 2019, pp. 975–980.
[9] A. Braun and A. Borrmann, "Combining inverse photogrammetry and bim for automated labeling of construction site images for machine learning," *Automation in Construction*, vol. 106, pp. 1–12, 2019.
[10] S. Zhou and W. Song, "Deep learning-based roadway crack classification using laser-scanned range images: A comparative study on hyperparameter selection," *Automation in Construction*, vol. 114, 2020.
[11] A. Sawhney, M. Riley, and J. Irizarry, *Construction 4.0: An innovation platform for the built environment*. Routledge, 2020.
[12] E. Monteiro, M. Oliveira, K. Almeida, J. Carvalho, T. Chaves, and E. ARIMATEIA, "Estudo da degradação nas marquises de edificações do centro histórico do recife." in *Congresso Internacional sobre Patología y Recuperación de Estructuras*, 2010.
[13] M. Plaisant, G. Almeida, and A. N. Haddad, "Patologias biológicas–tratamento de vida vegetal nos edifícios históricos do rio de janeiro iv cirmare–rio de janeiro," *IV Congresso Internacional na "Recuperação, Manutenção e Restauração de Edifícios (CIRMARE)*, 2015.
[14] F. R. de Assunção Rios, D. D. E. da Silva, J. N. da Costa, and B. J. S. Souza, "Análise das manifestações patológicas das marquises de concreto armado no centro de campina grande-pb," *Revista de Geociências do Nordeste*, vol. 5, pp. 12–22, 2019.
[15] M. Kaamin, N. Ahmad, S. Razali, M. Mokhtar, N. Ngadiman, D. Masri, I. Hussin, and L. Asri, "Visual inspection of heritage mosques using unmanned aerial vehicle (uav) and condition survey protocol (csp) 1 matrix: A case study of tengkera mosque and kampung kling mosque, melaka," vol. 1529, no. 3, 2020.
[16] M. LOUKMA and M. STEFANIDOU, "Causes of deterioration of ottoman mosques," *WIT Transactions on The Built Environment*, vol. 177, pp. 173–180, 2018.
[17] E. Rocha, J. Macedo, P. Correia, and E. Monteiro, "Adaptation of a damage map to historical buildings with pathological problems: Case study at the church of carmo in olinda, pernambuco," *Revista ALCONPAT*, vol. 8, no. 1, pp. 51–63, 2018.
[18] M. Längkvist, A. Kiselev, M. Alirezaie, and A. Loutfi, "Classification and segmentation of satellite orthoimagery using convolutional neural networks," *Remote Sensing*, vol. 8, no. 4, p. 329, 2016.
[19] G. Häufel, L. Lucks, M. Pohl, D. Bulatov, and H. Schilling, "Evaluation of cnns for land cover classification in high-resolution airborne images," in *Earth Resources and Environmental Remote Sensing/GIS Applications IX*, vol. 10790. International Society for Optics and Photonics, 2018, p. 1079003.
[20] E. Maltezos, A. Doulamis, N. Doulamis, and C. Ioannidis, "Building extraction from lidar data applying deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 1, pp. 155–159, 2018.
[21] M. C. Younis and E. Keedwell, "Semantic segmentation on small datasets of satellite images using convolutional neural networks," *Journal of Applied Remote Sensing*, vol. 13, no. 4, p. 046510, 2019.
[22] N. Mboga, T. Grippa, S. Georganos, S. Vanhuysse, B. Smets, O. Dewitte, E. Wolff, and M. Lennert, "Fully convolutional networks for land cover classification from historical panchromatic aerial photographs," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 167, pp. 385–395, 2020.
[23] H. Mohammadi and F. Samadzadegan, "An object based framework for building change analysis using 2d and 3d information of high resolution satellite images," *Advances in Space Research*, vol. 66, no. 6, pp. 1386–1404, 2020.
[24] P. Brazdil, C. G. Carrier, C. Soares, and R. Vilalta, *Metalearning: Applications to data mining*. Springer Science & Business Media, 2009.
[25] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2019, in press, available at http://automl.org/book.
[26] R. G. Mantovani, A. L. Rossi, E. Alcobaça, J. Vanschoren, and A. C. de Carvalho, "A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves svm classifiers," *Information Sciences*, vol. 501, pp. 193–221, 2019.
[27] A. L. C. Ottoni, E. G. Nepomuceno, M. S. de Oliveira, and D. C. R. de Oliveira, "Tuning of reinforcement learning parameters applied to sop using the scott–knott method," *Soft Computing*, vol. 24, p. 4441–4453, 2020.
[28] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," in *Deep learning in medical image analysis and multimodal learning for clinical decision support*. Springer, 2017, pp. 240–248.
[29] M. Kouzehgar, Y. K. Tamilselvam, M. V. Heredia, and M. R. Elara, "Self-reconfigurable façade-cleaning robot equipped with deep-learning-based crack detection based on convolutional neural networks," *Automation in Construction*, vol. 108, p. 102959, 2019.
[30] J. Giménez-Gallego, J. D. González-Teruel, M. Jiménez-Buendía, A. B. Toledo-Moreo, F. Soto-Valles, and R. Torres-Sánchez, "Segmentation of multiple tree leaves pictures with natural backgrounds using deep learning for image-based agriculture applications," *Applied Sciences*, vol. 10, no. 1, pp. 1–15, 2020.
[31] A. K. Rangarajan, R. Purushothaman, and A. Ramesh, "Tomato crop disease classification using pre-trained deep learning algorithm," *Procedia computer science*, vol. 133, pp. 1040–1047, 2018.
[32] O. Yadav, K. Passi, and C. K. Jain, "Using deep learning to classify x-ray images of potential tuberculosis patients," in *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2018, pp. 2368–2375.
[33] K. Bhosle and V. Musande, "Evaluation of deep learning cnn model for land use land cover classification and crop identification using hyperspectral remote sensing images," *Journal of the Indian Society of Remote Sensing*, vol. 47, no. 11, pp. 1949–1958, 2019.
[34] A. Wadhawan and P. Kumar, "Deep learning-based sign language recognition system for static signs," *Neural Computing and Applications*, pp. 1–12, 2020.
[35] S. Postalcıoğlu, "Performance analysis of different optimizers for deep learning-based image recognition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 34, no. 02, pp. 1–12, 2020.
[36] A. L. Majdik, C. Till, and D. Scaramuzza, "The zurich urban micro aerial vehicle dataset," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 269–273, 2017.
[37] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[38] D. C. Montgomery, *Design and analysis of experiments*. New York: John Wiley & Sons, 9th, 2017.
[39] J. Guo, Q. Wang, Y. Li, and P. Liu, "Façade defects classification from imbalanced dataset using meta learning-based convolutional neural network," *Computer-Aided Civil and Infrastructure Engineering*, 2020.

André Luiz Carvalho Ottoni possui graduação (2015) e mestrado (2016) em Engenharia Elétrica pela Universidade Federal de São João del-Rei (UFSJ). Atualmente é aluno de doutorado no Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal da Bahia (UFBA). Também é professor Assistente no Centro de Ciências Exatas e Tecnológicas (CETEC) da Universidade Federal do Recôncavo da Bahia (UFRB) e membro da Sociedade Brasileira de Automática (SBA). Tópicos de pesquisa: Inteligência Artificial, Aprendizado de Máquina, Otimização Combinatória e Robótica Inteligente.



Marcela Silva Novo possui graduação em Engenharia de Telecomunicações pela Universidade Federal Fluminense (2001), mestrado em Engenharia Elétrica pela Pontifícia Universidade Católica do Rio de Janeiro (2003) e doutorado em Engenharia Elétrica pela Pontifícia Universidade Católica do Rio de Janeiro (2007). De 2005 a 2006 foi pesquisadora visitante no ElectroScience Laboratory, The Ohio State University, USA. Atualmente é professora associada e vice-diretora da Escola Politécnica da UFBA. É membro da Diretoria Executiva da Sociedade Brasileira de Microondas e Optoeletrônica. Tópicos de pesquisa: eletromagnetismo computacional, métodos numéricos, análise e síntese de dispositivos de microondas e antenas, e processamento de sinais.