International Journal of Machine Learning and Cybernetics (2023) 14:171–186
https://doi.org/10.1007/s13042-022-01555-1

ORIGINAL ARTICLE

Tuning of data augmentation hyperparameters in deep learning to building construction image classification with small datasets

André Luiz C. Ottoni[1,2] · Raphael M. de Amorim[2] · Marcela S. Novo[3] · Dayana B. Costa[4]

Received: 24 May 2021 / Accepted: 22 March 2022 / Published online: 13 April 2022

Abstract
Deep Learning methods have important applications in the building construction image classification field. One challenge of this application is Convolutional Neural Networks adoption in a small datasets. This paper proposes a rigorous methodology for tuning of Data Augmentation hyperparameters in Deep Learning to building construction image classification, especially to vegetation recognition in facades and roofs structure analysis. In order to do that, Logistic Regression models were used to analyze the performance of Convolutional Neural Networks trained from 128 combinations of transformations in the images. Experiments were carried out with three architectures of Deep Learning from the literature using the Keras library. The results show that the recommended configuration (Height Shift Range = 0.2; Width Shift Range = 0.2; Zoom Range =0.2) reached an accuracy of 95.6% in the test step of first case study. In addition, the hyperparameters recommended by proposed method also achieved the best test results for second case study: 93.3%.

Keywords Deep learning · Convolutional neural networks · Hyperparameter tuning · Data augmentation · Building construction image classification

1 Introduction

Deep Learning methods have important applications in the Digital Image Processing field [4, 23, 42, 47]. In this sense, a possible application of Deep Learning is building construction area [8, 10, 14, 41, 53]. In recent literature, there are several applications in this research field, such as: crack detection [8, 54], road crack classification [53], safety guardrail detection [22], structural damage recognition [11], detecting safety helmet [41], safety harness detection [10], classification of rock fragments [50], damage detection of a steel bridge [1], tunnel lining defects [49] and facade defects classification [14].

Deep Learning methods can also be applied in recognition of vegetation in building facades images [32]. In fact, the growth of biological manifestations on building facades may indicate the deterioration and degradation of constructions [2, 24]. In addition, the detection of this pathology in inspection images can assist in the conservation of historic buildings [7, 21, 24, 37]. In this sense, [32] proposes a Deep Learning approach for recognizing vegetation in buildings. Another possibility is to use Deep Learning analysis of roof structures [33]. In the literature, there are several examples of works that investigated the efficient of roofs structure [5, 12, 33, 43]. For example, in a recent study, [33] proposes a methodology to tuning of two hyperparameters (learning rate and optimizer) of Neural Networks in the building roof image classification. It is also worth noting that, one of the relevant factors on [32] and [33] was the experiments with Data Augmentation. [32] verified the improvement

✉ André Luiz C. Ottoni
andre.ottoni@ufrb.edu.br

Raphael M. de Amorim
amorimba@gmail.com

Marcela S. Novo
marcela.novo@ufba.br

Dayana B. Costa
dayanabcosta@ufba.br

1 Technologic and Exact Center, Federal University of Recôncavo da Bahia, Cruz das Almas, Brazil

2 Electrical Engineering Graduate Program, Federal University of Bahia, Salvador, Brazil

3 Department of Electrical and Computer Engineering, Federal University of Bahia, Salvador, Brazil

4 Department of Structural and Construction Engineering, Federal University of Bahia, Salvador, Brazil

in validation accuracy when using Data Augmentation to increase the training database.

In fact, Data Augmentation techniques play an important role in the application of Machine Learning in small datasets [4, 9, 42, 51, 52]. This is because, the generation of artificial images directly contributes to increase the capacity for the generalization of the Deep Learning model and thus decrease the chance of overfitting [4, 9]. In this respect, one of the challenges of using Data Augmentation is the definition of which transformations (such as zoom, rotation, flip) will be applied to the images [6, 28, 34, 44, 48]. In terms of Machine Learning, this problem can be treated as in the area of Hyperparameter Tuning [19, 20, 27, 30, 31, 39, 40].

In the literature, some studies have analyzed the influence of Data Augmentation hyperparameter combinations in different applications, such as: plant classification [34], transmission line inspection [44] and covid-19 diagnostic process in chest X-ray radiological imaging [28]. In [48], different types of Data Augmentation methods were analyzed for crack detection in constructions. However, the literature lacks proposals to optimize the combinations of Data Augmentation hyperparameters for the application of Deep Learning in building construction image classification, especially in the recognition of vegetation on building facades and roofs defects classification.

The objective of this paper is to propose a rigorous methodology for tuning of Data Augmentation hyperparameters in Deep Learning to building construction image classification with small data sets. For this, two case studies are observed: vegetation recognition in facades [32] and roofs structures analysis [33]. In order to do that, Logistic Regression models [16] will be used to analyze the performance of Convolutional Neural Networks (CNN) [4, 9] trained from 128 combinations of transformations in the images. For comparison purposes, three CNN architectures from the literature will also be adopted: MobileNet [17], DenseNet-121 [18] e CNN8 [32].

This paper is organized into five sections. Section 2 presents theoretical concepts of CNNs and Logistic Regression. Section 3 presents the proposed methodology. Sections 4 and 5 describe the results and conclusions, respectively.

2 Theoretical foundation

2.1 Convolutional neural networks

Convolutional neural networks (CNNs) are Deep Learning methods with several researches in computer vision field [4, 9, 15, 23]. One of the main factors that make CNNs a relevant Machine Learning technique is the ability to automatically extract features from processed images [9]. In addition, another important point is the use of layers and elements with different functionalities in the network architecture, such as [4, 9]:

- Input layer: receives input signals (e.g .: image).
- Weights: adjusted during the training process (trainable parameters).
- Convolutional filters (kernels): have a set of weights, according to their size. For example, if the kernel size is 3×3, then the filter contains 9 trainable weights.
- Activation function: transforms a signal into a limited output. Some examples are the functions ReLu, softmax and sigmoid.
- Convolutional Layer: applies the convolution operation between the filters and the input matrix in the layer. As an output, new matrices are generated (feature map), according to the number of kernels in the layer.
- Pooling: applies a transformation to decrease the input matrix dimensions. For this, statistical functions can be used: maximum (max) or average (avg).
- Flatten: transforms the matrices resulting from convolutional operations into a single vector.
- Dropout: randomly disconnects a set of neurons at each training epoch.
- Fully connected layers: similar to the structures of traditional Artificial Neural Networks, in which, all neurons and layers are connected.
- Output layer: shows the output of CNN, such as a binary classifier neuron.

Thus, in view of the complexity of the CNN architectures, an important factor is the use of tools for efficient implementation [4, 9]. In this line, it is worth mentioning the Keras library [4]. Keras is available on R interface[1] and Python language for development of Deep Learning applications. In addition, it allows execution on CPU or GPU. Another relevant factor is the simplicity to use Data Augmentation methods. In this sense, the Keras library was adopted in this work, as described in Sect. 3.

2.2 Logistic regression

The methods based on Linear Regression (simple and multiple) aim to model a continuous output from one or more independent variables [29]. On the other hand, Logistic Regression is a technique addressed for the analysis of categorical data [13, 16]. Moreover, in a logistic function, the variable response is binary or dichotomous.

The Logistic Regression model can be represented by Eq. (1) [13]:

[1] https://keras.rstudio.com/.

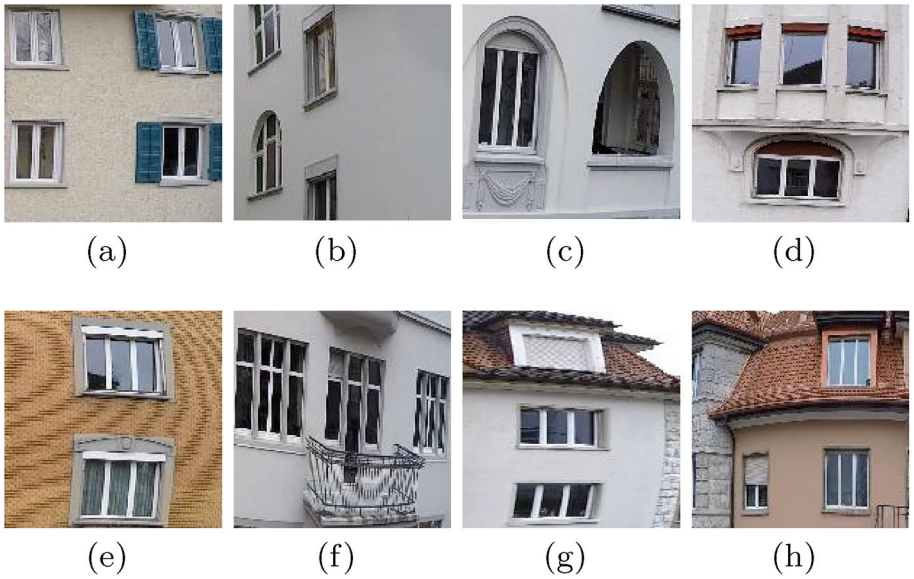

Fig. 1 Examples of images of the class 0 - without vegetation on the facade

$$p(x) = \frac{exp(\beta_0 + \sum_{k=1}^{m} \beta_k x_k)}{1 + exp(\beta_0 + \sum_{k=1}^{m} \beta_k x_k)} \quad (1)$$

where, β_0 to β_k are the coefficients of the regression model; x are the independent variables; and $p(x)$ is the probability of success. Thus, if $p(x)$ is the probability of an event occurring, then the expression $1 - p(x)$ represents the probability of an event not occurring. The ratio between $p(x)$ and $1 - p(x)$ is called *chance* (Eq. (2)) [13]:

$$chance = \frac{p(x)}{1 - p(x)} \quad (2)$$

In this line, the neperian logarithm of *chance* provides a linear model, according to Eq. (3) [13]:

$$ln\left[\frac{p(x)}{1 - p(x)}\right] = \beta_0 + \sum_{k=1}^{m} \beta_k x_k \quad (3)$$

where, this equation is called logit and is a simplification of the Logistic Regression model.

3 Methodology

3.1 Database of the first case study

In this study, the database presented by [32] was used for training, validation and testing of Convolutional Neural Networks in the recognition of vegetation on building facades. The dataset has 390 images, divided into two classes:

1. Class 0: without vegetation on the building's facade.
2. Class 1: with vegetation on the building's facade.

According to [32], the images of the training and validation datasets were defined from photographs adapted from The Zurich Urban Micro Aerial Vehicle Dataset [26]. These images were recorded in 2015 by an vant from the urban streets of Zurich (Switzerland). On the other hand, the images of the test dataset were selected from the website Pixabay.[2] The dataset analyzed during the current study are available from the corresponding author on request or in the web link:

– drive.google.com/file/d/1l6KA80mZdKqxlpfpenIH57mCYESL3uyq/

Figures 1 and 2 present examples of images from the database of classes 0 and 1, respectively.

The database (390 images) was partitioned following the same structure proposed by [32]:

– Training (250 images): 125 images in class 0 (without vegetation) and 125 images in class 1 (with vegetation).
– Validation (50 images): 25 images in class 0 (without vegetation) and 25 images in class 1 (with vegetation).
– Test (90 images): 45 images in class 0 (without vegetation) and 45 images in class 1 (with vegetation).

3.2 Data augmentation

Data preprocessing is an important stage in the Machine Learning field [9]. This is because, this step can decrease the learning complexity and improve the accuracy results

[2] www.pixabay.com.

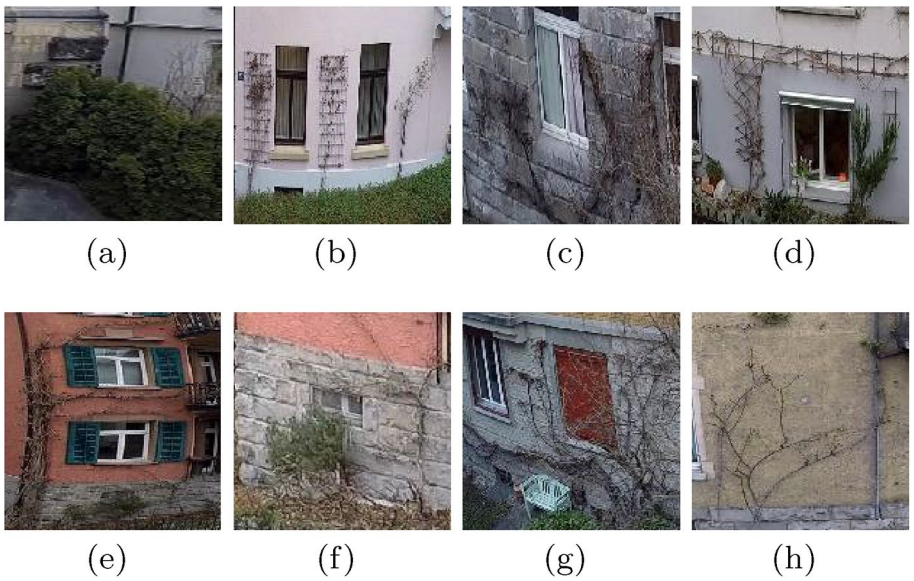

Fig. 2 Examples of images of the class 1 - with vegetation on the facade

[9]. In this line, Data Augmentation techniques can be used in the training of CNNs [4, 9, 42, 51, 52].

Data Augmentation is an approach applied mainly to small data learning [4, 42]. In this sense, Data Augmentation methods generate more data for training from the existing images [4]. The aim is to increase the CNN model's ability to generalize and avoid overfitting [4, 9]. For this, artificial images are created from random transformations in the original data [4]. Zoom, rotation and flip are some examples of possible transformations for the generation of augmented images [9].

In this paper, Data Augmentation methods were applied from Keras library in R software [4]. For this, a *image_data_generator()* method was adopted [4]. The function *image_data_generator()* generates batches of data with new modified images from the original data. In this regard, the following random transformations[3] were used to increase training data [4]:

- **Rotation range**: an integer number that defines the degree range for random rotations. Rotation is a circular movement around a fixed point. The processing images will have random rotations on a predefined range of degrees according with data entrance.
- **Horizontal flip**: if this input is"true"them the images will be randomly mirrored in the horizontal direction (left-right).
- **Vertical flip**: if this input is"true"them the images will be randomly mirrored in the vertical direction (up-bottom).
- **Shear range**: distort the image along an axis to create or rectify the perception angles. There are two shear transformation, X-Shear that shift X coordinates values and Y-Shear that's shift Y coordinates values.
- **Width shift range**: shifts the image randomly to the left or to the right (horizontal shifts). If the value is float and ≤ 1 it will take the percentage of total width as range. For example, in an image that width is 100 pixels and if width_shift_range = 1.0 then it will shift image randomly between -100% to 100% or -100px to 100px. Positive values will shift the image to the right side and negative values will shift the image to the left side.
- **Height shift range**: shifts the image randomly to up or down (vertically shifts). If the value is float and ≤ 1 it will take the percentage of total height as range. For example, in an image that height is 100 pixels and if height_shift_range = 1.0 then it will shift image randomly between -100% to 100% or -100px to 100px. Positive values will shift the image to the upside and negative values will shift the image to underside.
- **Zoom range**: it will do a randomly augmentation of the image adding new pixels values. It can be specified with the percentage of the zoom as single float or a range as an array. For example, if zoom_range = 0.4 the range will be [0.6, 1.4] between 60% (zoom in) or 140% (zoom out).

Figure 3 presents examples of images generated by Data Augmentation with the Keras library.

The number of artificially generated images depends on training settings: *batch_size*, *steps_per_epoch* and *epoch*. For example, in this study, these parameters were defined in first phase of experiments as: *batch_size* = 32; *steps_per_epoch* = 100; and *epoch* = 10. Thus, for each simulation were generated randomly around 32,000 new images for training. This value is more than 100 times greater than the number of original photographs for training (250).

[3] https://keras.rstudio.com/reference/.

Fig. 3 Examples of images generated by Keras data augmentation: **a** original image; **b–d** rotation range ($R = 40$); **e** horizontal flip ($H = TRUE$); **f** vertical flip ($V = TRUE$); **g** and **h** height shift range ($He = 0.2$); **l** shear range ($S = 0.2$); **j–l** width shift range ($W = 0.2$); **m–p** zoom range ($Z = 0.2$)

3.3 Neural network architectures

In this paper, three CNNs architectures were adopted: CNN-8 [32], DenseNet-121 [18] and MobileNet [17]. Recently, these structures (or variations) are discussed in some papers in research field of building construction image processing with deep learning, such as, in the tasks: crack detection (DenseNet) [46], structural health monitoring (FC-DenseNet) [38], detecting safety helmet (SSD MobileNet) [41], road damage detection (SSD MobileNet) [25] and recognition of vegetation in buildings (CNN-8) [32].

In this study, CNN architectures were used for binary classification, for example, class 0 (without vegetation on the building's facade) and class 1 (with vegetation on the building's facade) [32]. For this, the *keras_model_sequential()* method in the Keras library was used [4], as described below:

- **CNN-8**: CNN architecture used by [32] to vegetation image recognition in buildings. The structure has 8 layers and 3,985,345 trainable parameters. In addition, this architecture is based on a model proposed by [4], originally with 12 layers.
- **DenseNet-121**: Dense Convolutional Network is an architecture proposed by [18]. This structure is characterized by connecting each layer to all other layers (dense connection). Moreover, it has 7,479,169 trainable parameters. To use this architecture, the *application_densenet121()* method in the Keras library was adopted.

– **MobileNet**: CNN architecture proposed by [17] for mobile and embedded vision applications. The structure uses deptwise separable convolutions (factorized convolutions). In addition, it has 28 layers and 3,732,289 trainable parameters. To use this architecture, the *application_mobilenet()* method in the Keras library was adopted.

In all experiments, CNN architectures were trained with an adagrad optimizer and a learning rate of 0.01. In addition, the dimensions $50 \times 50 \times 3$ were standardized as input to the neural network: *input_shape = c(50, 50, 3)*. It is also noteworthy that all three architectures were configured with the last two layers as fully connected. In the last layer having the binary classifier neuron with sigmoid activation function [4].

3.4 Hyperparameter tuning

3.4.1 Design of experiments

In this section, the design of experiments for tuning of data augmentation hyperparameters with Logistic Regression [16] is described. The simulations of the Convolutional Neural Network models were conducted in the R software [35] with the Keras library [4]. For this, an Intel Core i7-8565 (CPU) and NVIDIA GeForce MX110 (GPU) were used.

For the experiments evaluation were used three metrics: accuracy in validation or testing (Acc), number of images correctly classified (C) and number of images incorrectly classified, that is, errors (E). Equations (4) to (6) present these formulas.

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}, \tag{4}$$

$$C = TP + TN, \tag{5}$$

$$E = FP + FN, \tag{6}$$

where,

– TP: true positives, that is, correct classifications in class 1 (facade with vegetation).
– FN: false negatives, that is, incorrect classifications in class 1 (facade with vegetation).
– TN: true negatives, that is, correct classifications in class 0 (facade without vegetation).
– FP: false positives, that is, incorrect classifications in class 0 (facade without vegetation).

The experiments were conducted in three stages:

1. Data Augmentation Hyperparameters.
2. Data Augmentation and CNN Architectures.
3. Test Experiments.

In the first phase, seven hyperparameters were defined for adjustment, each with two levels of treatments (0 - without transformation and 1 - with transformation):

– Rotation Range (R): 0 or 40.
– Horinzontal Flip (H): FALSE or TRUE.
– Vertical Flip (V): FALSE or TRUE.
– Height Shift Range (He): 0 or 0.2.
– Shear Range (S): 0 or 0.2.
– Width Shift Range (W): 0 or 0.2.
– Zoom Range (Z): 0 or 0.2.

Thus, a total of 128 (2^7) combinations of data augmentation hyperparameters were analyzed in the first stage. For each configuration, five CNN models (repetitions) were trained in 10 epochs with 100 steps (*steps por epoch*) adopting the MobileNet architecture [17]. The metrics observed in this phase were the total number of images correctly classified (C) and errors (E) in the validation dataset, used to adjust a Logistic Regression model.

In the second stage of experiments, the best combinations of the first phase were used. In addition, three CNN architectures from the literature were adopted: MobileNet [17], DenseNet-121 [18] e CNN8 [32]. For each combination (configuration of data augmentation × architecture), 5 repetitions were performed with 20 epochs. The total correct classifications (C) and errors (E) were observed in the validation dataset. Furthermore, the accuracy (Acc) in the validation step was also analyzed.

Finally, in the third phase of experiments, the hyperparameter combinations performance were analyzed in the test dataset. In this sense, new trainings were carried out with the data augmentation configurations selected in 5 repetitions with 30 epochs. For each of the CNN models trained in this phase, the accuracy in the classification of the test database was analyzed.

3.4.2 Logistic regression method

In this paper, the method for hyperparameter tuning uses Logistic Regression [16]. The objective is to evaluate the probability of hits and errors in the building construction image classification, according to the settings of data augmentation. For this, the response variable (y) is binary and was modeled as follows:

$$y = \begin{cases} 1: & \text{correct image classification}. \\ 0: & \text{incorrect image classification}. \end{cases}$$

For the first step, the explanatory variables (x_1, x_2, x_3, x_4, x_5, x_6 and x_7) refer to the seven hyperparameters analyzed:

$$\begin{cases} x_1 : \text{Rotation Range (R)} \\ x_2 : \text{Horizontal Flip (H)} \\ x_3 : \text{Vertical Flip (V)} \\ x_4 : \text{Height Shift Range (He)} \\ x_5 : \text{Shear Range (S)} \\ x_6 : \text{Width Shift Range (W)} \\ x_7 : \text{Zoom Range (Z)} \end{cases}$$

The Equation (7), in turn, presents the Logistic Regression model (logito format) proposed for the recommendation of hyperparameters:

$$ln\left[\frac{p(x)}{1-p(x)}\right] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \ldots\ldots + \beta_4 x_4 + \beta_5 x_5 + \beta_6 x_6 + \beta_7 x_7 \quad (7)$$

The coefficients (β) of Eq. (7) can be obtained by the maximum likelihood method [13]. Then, the regression coefficients hypothesis test must be performed. In this sense, the significance of the effects of each variable present in the model is analyzed in two hypotheses:

$$\begin{cases} H_0 : \beta_k = 0, \\ H_a : \beta_k \neq 0. \end{cases}$$

When the initial hypothesis (H_0) is accepted ($p > 0.05$), the variable x_k (associated with the β_k coefficient) does not have statistical significance in the model. On the other hand, if the alternative hypothesis is accepted ($p < 0.05$), the hyperparameter (k) has significance in the Logistic Regression model.

The adjusted coefficients (β) also may perform the calculations of the odds associated with each hyperparameter configuration. In this aspect, the OR metric represents the odds ratio of correct classification between the two levels of a hyperparameter. For example, the odds ratio for hyperparameter 1 (*Rotation Range*) is given by Eq. (8):

$$OR_1 = exp(\beta_1), \quad (8)$$

where OR_1 is the odds ratio of level 1 ($R = 40$) in relation to level 0 ($R = 0$) of hyperparameter 1 (*Rotation Range*). Thus, if $OR_1 > 1$ the chance of success in adopting $R = 40$ is greater than $R = 0$. Otherwise ($OR_1 < 1$), then the chance of CNN correctly classifying an image is greater if trained without the rotation transformation. A similar analysis can be made after calculating the odds ratio of the other analyzed hyperparameters. Thus, Eq. () presents the general formulation for the odds ratio:

$$OR_k = exp(\beta_k). \quad (9)$$

Therefore, odds ratio indices are used to define hyperparameter configurations for the sequence of experiments, as will be described in the next subsection (HPtuningLogReg Algorithm).

In the sequence, logistic regression models are also used to analyze the results of the second stage of experiments. In this case, the objective is to evaluate the influence of the selected hyperparameter combinations for the three CNN architectures adopted (CNN8 [32], DenseNet-121 [18] and MobileNet [17]). For this, three logistic regression models are adjusted (one per architecture) and the odds ratio indices for the hyperparameter configurations are observed.

3.4.3 HPtuningLogReg algorithm

Algorithm 1 presents the method proposed in R language for tuning of data augmentation hyperparameters with Logistic Regression: HPtuningLogReg Algorithm. The code has been divided into four steps: data input, adjustment of the logistic regression model, hyperparameter tuning and summary.

In the first phase (lines 1 to 14), the results of the experiments are read and prepared for the method sequence. The"correct"and "error"vectors store the number of correct and incorrect classifications, respectively, for each of the observed hyperparameter configurations.

Then, in line 16, the function *glm* of the R language is used to adjust the Logistic Regression model. In addition, the *anova* method (line 17) is also used to perform statistical tests of analysis of variance and to calculate the $p - value$ ("paov"). In sequence, from the adjusted coefficients ("modelglm$coefficients"), the odds ratio measures are calculated (line 18).

In phase 3 (lines 19 to 45), the Logistic Regression model is adopted to hyperparameter tuning. For this, a repetition loop is performed by varying the hyperparameter index (data augmentation transformation type): 1 - *R*, 2 - *H*, 3 - *V*, 4 - *He*, 5 - *S*, 6 - *W* and 7 - *Z*. In line 21, the statistical significance of the variable present in the model is analyzed. If the alternative hypothesis (H_1) is accepted ("paov$'Pr(>Chi)'[i+1]< 0.05") there is significance for the coefficient β_k, that is, there is a statistical difference between the two treatments of the hyperparameter k. In this case, the value of the odds ratio is presented and then recommended hyperparameter level with the greatest chance of success in the validation dataset image classification (lines 22 to 35). On the other hand, if initial hypothesis (H_0) is accepted (lines 36 to 45), there is no statistically significant difference between the two values of the hyperparameter k ($p > 0.05$). Finally, default treatment ("H[i,2]") is recommended, that is, the transformation k in the data augmentation process should not be applied.

In step 4, a summary of the recommended hyperparameter values is presented. In this case, only the hyperparameters

whose decision variables received level 1 (C[i] == 1) in step 3 are shown.

```
# Step 1: Data Input
dat <- read.delim("data.txt")
attach(dat)
dadosf<-
  as.data.frame(cbind(correct,error,R,H,V,He,S,W,Z))
attach(datf)
h1<-c("Hyper1 - Rotation Range (R)", "0", "40")
h2<-c("Hyper2 - H. Flip (H)", "FALSE", "TRUE")
h3<-c("Hyper3 - V. Flip (V)", "FALSE", "TRUE")
h4<-c("Hyper4 - Height Shift R. (He)", "0", "0.2")
h5<-c("Hyper5 - Shear Range (S)", "0", "0.2")
h6<-c("Hyper6 - Width Shift R. (W)", "0", "0.2")
h7<-c("Hyper7 - Zoom Range (Z)", "0", "0.2")
h<-rbind(h1, h2, h3, h4, h5, h6, h7)
C<-c(0,0,0,0,0,0,0)
# Step 2: Logistic Regression Model
modelglm<-
  glm(cbind(strtoi(correct),strtoi(error))~
  R+H+V+He+S+W+Z,
  family=binomial(link="logit"), data=datf)
paov<-anova(modelglm, test="Chisq")
OR <- exp(modelglm$coefficients)
# Step 3: Hyperparameter Tuning
for (i in 1:7){
  if(paov$`Pr(>Chi)`[i+1]< 0.05){
    print("Hyperparameter Tuning Reglog")
    print(h[i])
    print("With statistical significance (p < 0.05)")
    print("Beta:")
    print(modelglm$coefficients[i+1])
    print("Odds Ratio:")
    print(OR[i+1])
    print("Recommended hyperparameter:")
    if (OR[i+1]<1){
      print(h[i,2])
    }else{
      print(h[i,3])
      C[i]<-1
    }
  }else{
    print("Hyperparameter Tuning Reglog")
    print(h[i])
    print("No statistical significance (p > 0.05)")
    print("Beta:")
    print(modelglm$coefficients[i+1])
    print("Recommended hyperparameter:")
    print(h[i,2])
  }
}
# Step 4: Hyperparameter Tuning - Summary
for (i in 1:7){
    if (C[i] == 1){
      print("Hyperparameter Tuning - Summary")
      print(h[i])
      print(h[i,3])
    }
}
```

Algorithm 1: HPtuningLogReg in R language.

3.5 Hyperparameter tuning to second small dataset

In this case study, another type of problem in buildings was analyzed: gutter integrity and cleanliness pathology in roofs [43]. For this, images were used from the database presented and described by [33, 43, 45] and made available by the Research Group in Construction Technology and Management (School of Engineering - UFBA).[4] These images were captured from roof inspections with an unmanned aerial vehicle.

In a previous study, [33] used this dataset in experiments to tuning of two CNN hyperparameters (learning rate and optimizer). For this, the images were divided into two classes: (0) roofs with clean gutters and (1) roofs with dirty gutters. Thus, the database adopted by [33] has 220 images, separated for the training, validation and test phases:

- Training (160 images): 80 images in class 0 and 80 images in class 1.
- Validation (30 images): 15 images in class 0 and 15 images in class 1.
- Test (30 images): 15 images in class 0 and 15 images in class 1.

Figures 4 and 5 present examples of images (two classes) of the second small dataset.

In this sense, the Hyperparameter Tuning methodology (Section 3.4) was adopted in new experiments with this second case study. Then, HPtuningLogReg was applied to tuning of Data Augmentation hyperparameters.

The dataset analyzed during the second case study is available from the corresponding author on request or in the web link:

- https://abre.ai/dataset2

4 Results

4.1 Results of first small dataset

This section presents the results for the first case study: recognition of vegetation on building facades.

4.1.1 Stage 1: Data augmentation hyperparameters

In stage 1, HPtuningLogReg algorithm was adopted to adjust the logistic regression model and tuning of Data Augmentation hyperparameters. For this, results of 128

[4] getec.eng.ufba.br.

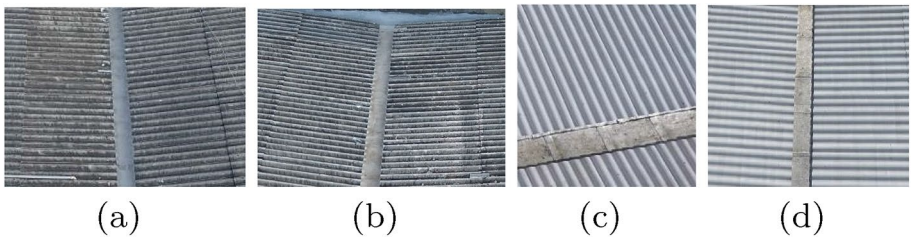

Fig. 4 Examples of images of the class 0 (roofs with clean gutters) in second small dataset

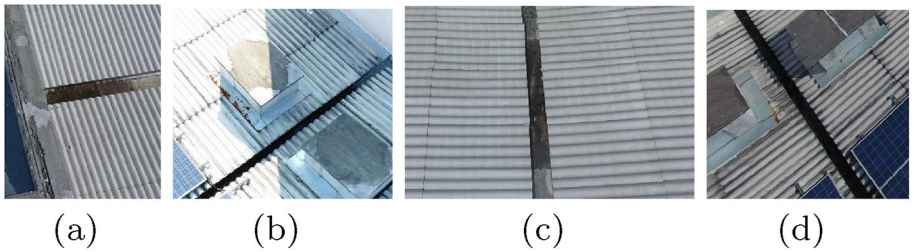

Fig. 5 Examples of images of the class 1 (roofs with dirty gutters) in second small dataset

Table 1 Results of Data Augmentation hyperparameter tuning with logistic regression in stage 1.

Hyperparameter	β	p	*Value*	x	*OR*
Rotation R. (R)	– 0.005	0.86	0	0	0.995
Hor. Flip (H)	– 0.118	0.00	False	0	0.888
Vertical Flip (V)	– 0.130	0.00	False	0	0.878
Height S. R. (He)	0.179	0.00	0.2	1	**1.196**
Shear Range (S)	– 0.008	0.79	0	0	0.992
Width S. R. (W)	0.114	0.00	0.2	1	**1.121**
Zoom Range (Z)	0.111	0.00	0.2	1	**1.118**

Bold values indicate the hyperparameters with $OR > 1$ and $p < 0.05$

hyperparameters combinations analyzed were used. The Equation (10) presents the adjusted linear model (logito).

$$\begin{aligned} y =& \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdot \\ & \cdot + \beta_4 x_4 + \beta_5 x_5 + \beta_6 x_6 + \beta_7 x_7 \\ =& 1.524 - 0.005 x_1 - 0.118 x_2 - 0.130 x_3 + \cdot \\ & \cdot + 0.179 x_4 - 0.0085 x_5 + 0.114 x_6 + 0.111 x_7 \end{aligned} \tag{10}$$

Table 1 shows the results of the test statistic (p), recommended values and odds ratio (OR) per hyperparameter.

The results of Table 1 shows that the effects related to transformations of *Rotation Range* and *Shear Range* have no statistical effect ($p > 0.05$). Thus, the algorithm recommended the use of the default value for these hyperparameters ($R = 0$ and $S = 0$). On the other hand, the effects of the variables referring to *Horizontal Flip* and *Vertical Flip* showed statistical significance ($p < 0.05$). However, the odds ratio values for these hyperparameters were less than 1 ($OR < 1$). Thus, HPtuningLogReg algorithm recommended the use of $H = 0$ and $V = 0$.

Table 1 also presents the results for the other three transformations: *Height Shift Range*, *Width Shift* and *Zoom*. The effects of these variables were statistically significant ($p < 0.05$). The recommended value of *Height Shift* method was 0.2 with an estimated odds ratio of 1.196. In this respect, the adjusted model reveals that adopting the level $x_{He} = 1$ has around 20% more chances of success in the image classification, in relation to not adopting this transformation in the training base. The adjusted values for *Width Shift* and *Zoom* were also 0.2, with odds ratios of 1.121 and 1.118, respectively. Thus, it is estimated that the chance of correct image classification when using $W = 0.2$ or $Z = 0.2$ is around 12% greater than performing the training without these Data Augmentation effects.

Thus, from the Logistic Regression results, the HPtuningLogReg algorithm recommended three transformations in the images for the training process: *Height Shift Range*, *Width Shift Range* e *Zoom Range*. These hyperparameters analyzed at two levels each, result in eight combinations of Data Augmentation transformations ($2 \times 2 \times 2$). Table 2 presents these combinations set and their respective levels of decision variables

The hyperparameter combinations presented in Table 2 were used in the next stage of experiments, as shown in the following section.

4.1.2 Stage 2: Data augmentation and CNN architectures

In stage 2, the hyperparameter combinations defined in the previous phase were evaluated in conjunction with three architectures in the literature: CNN8 [32], DenseNet-121 [18] and MobileNet [17]. Table 3 presents the results of accuracy in the validation step and the statistical metrics of the logistic regression models (OR and p).

From the Table 3 it is possible to observe that the highest accuracy average (87.6%) for the CNN8 architecture was

Table 2 Hyperparameter combinations of data augmentation selected in stage 1

Comb.	He	W	Z	x_{He}	x_W	x_Z
1	0	0	0	0	0	0
2	0	0	0.2	0	0	1
3	0	0.2	0	0	1	0
4	0	0.2	0.2	0	1	1
5	0.2	0	0	1	0	0
6	0.2	0	0.2	1	0	1
7	0.2	0.2	0	1	1	0
8	0.2	0.2	0.2	1	1	1

Table 3 Results of validation accuracy (%) and statistical metrics for each method (CNN architecture + data augmentation combination)

Arch.	Comb.	1	2	3	4	5	Mean	OR	p
	1	78.0	78.0	78.0	76.0	82.0	78.4	1.000	–
	2	86.0	82.0	78.0	78.0	76.0	80.0	1.102	0.66
	3	86.0	82.0	86.0	86.0	84.0	84.8	1.537	0.07
CNN8	4	92.0	88.0	86.0	84.0	92.0	88.4	2.099	**0.00**
	5	84.0	90.0	80.0	86.0	86.0	85.2	1.586	**0.05**
	6	84.0	84.0	84.0	86.0	82.0	84.0	1.445	0.11
	7	90.0	92.0	88.0	90.0	88.0	89.6	2.374	**0.00**
	8	88.0	88.0	86.0	90.0	86.0	87.6	1.946	**0.01**
	1	88.0	84.0	90.0	88.0	84.0	86.8	1.000	–
	2	90.0	86.0	94.0	92.0	84.0	89.2	1.256	0.41
	3	92.0	92.0	88.0	90.0	86.0	89.6	1.310	0.33
DenseNet-121	4	90.0	88.0	86.0	90.0	92.0	89.2	1.256	0.41
	5	86.0	90.0	94.0	88.0	86.0	88.8	1.206	0.49
	6	90.0	88.0	90.0	88.0	92.0	89.6	1.310	0.33
	7	90.0	94.0	92.0	92.0	90.0	91.6	1.658	0.09
	8	92.0	96.0	96.0	92.0	94.0	94.0	2.382	**0.01**
	1	76.0	74.0	76.0	78.0	76.0	76.0	1.000	–
	2	84.0	88.0	88.0	88.0	90.0	87.6	2.231	**0.00**
	3	82.0	84.0	90.0	82.0	78.0	83.2	1.564	**0.05**
MobileNet	4	92.0	88.0	90.0	88.0	90.0	89.6	2.720	**0.00**
	5	82.0	80.0	78.0	84.0	82.0	81.2	1.364	0.16
	6	84.0	86.0	86.0	86.0	90.0	86.4	2.006	**0.00**
	7	88.0	92.0	88.0	86.0	94.0	89.6	2.721	**0.00**
	8	92.0	90.0	90.0	92.0	90.0	90.8	3.117	**0.00**

Bold values indicate the data augmentation combinations with $p < 0.05$

achieved by adopting the combination 7 ($He = 0.2; W = 0.2; Z = 0$). In this case, adopting configuration 7 has approximately 2 times more chances of success ($OR = 2.374$) in the classification in relation to the reference combination ($He = 0; W = 0; Z = 0$). It is also noteworthy that the four combinations (4, 5, 7 and 8) showed statistical significance ($p \leq 0.05$). On the other hand, when analyzing the results of DenseNet-121 in the Table 3, the highest mean accuracy (94.0%) was obtained by the combination 8. In addition, only this configuration ($He = 0.2; W = 0.2; Z = 0.2$) was statistically significant (level of 5%). The experiments with the MobileNet architecture, revealed that adopt the configuration 8 has around 3 times more chances of success ($OR = 3.117$) in image classification. Moreover, five combinations (2, 3, 5, 6, 7 and 8) are statistically different ($p \leq 0.05$).

In this sense, configuration 8 ($He = 0.2$; $W = 0.2$; $Z = 0.2$) was the only to present statistical significance for the three architectures. In addition, the odds ratio for this combination was over 1.9 for CNN8, DenseNet-121 and MobileNet. Thus, combination 8 was selected for the sequence of experiments in the test stage.

To illustrate, Figs. 6 and 7 present samples of images generated by Data Augmentation, adopting the combination 8: $He = 0.2; W = 0.2; Z = 0.2$.

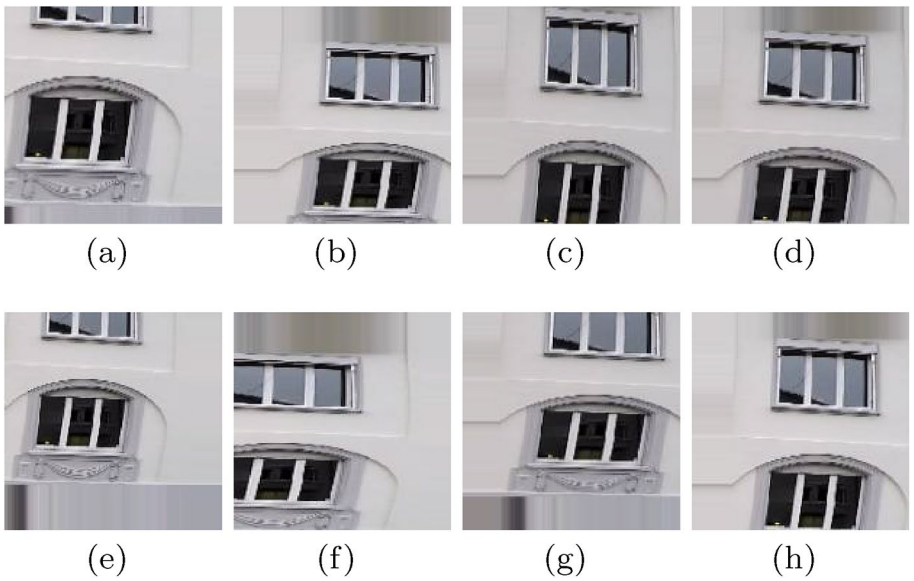

Fig. 6 Examples of images generated by data augmentation ($He = 0.2; W = 0.2; Z = 0.2$) for class 0 (without vegetation on the building facade)

Fig. 7 Examples of images generated by data augmentation ($He = 0.2; W = 0.2; Z = 0.2$) for class 1 (with vegetation on the building facade)

Table 4 Maximum accuracy in the test step in each of the repetitions and respective mean accuracy (M) for first small dataset

Arch.	C.	1	2	3	4	5	M.
CNN8	P	**95.6**	87.8	92.2	93.3	91.1	**92.0**
CNN8	L	91.1	80.0	93.3	91.1	93.3	89.8
DenseNet	P	77.8	77.8	73.3	83.3	65.6	75.6
DenseNet	L	87.8	77.8	83.3	71.1	70.0	78.0
MobileNet	P	71.1	65.6	81.1	74.4	63.3	71.1
MobileNet	L	54.4	87.8	58.9	55.6	53.3	62.0

Bold values indicate the best result of test accuracy and mean accuracy

Comparison between methods recommended by the data augmentation configurations (Proposed (P) and Literature (L)) and architectures

Table 5 Confusion matrix with the best results for the test step (first small dataset)

TP = 43	$FN = 2$
$FP = 2$	**TN = 43**

Bold values indicate the number of true positives (TP) and true negatives (TN)

Table 6 Selected hyperparameters for first small dataset

Hyperparameter	Recommendation
Architecture	CNN8
Height Shift Range (He)	0.2
Width Shfit Rang (W)	0.2
Zoom Range (Z)	0.2

4.1.3 Tests results

In this step, simulations were performed out on the test dataset adopting three architectures (CNN8, DenseNet-121 and MobileNet) and two data augmentation configurations:

- Proposed in this paper (P) - defined from steps 1 and 2 (Hyperparameter Tuning): ($He = 0.2; W = 0.2; Z = 0.2$).
- Literature (L) - presented in [4] and used by [32] for the same dataset of this study: ($R = 40$; $H = TRUE$; $He = 0.2$; $S = 0.2$; $W = 0.2$; $Z = 0.2$).

Table 4 presents the accuracy in test step for each analyzed configurations.

From the Table 4 it is possible to observe that the highest accuracy average (92.0%) was achieved by the CNN8 architecture when adopting the proposed data augmentation combination. Moreover, this configuration (CNN8 + P) also resulted in the highest accuracy value in one repetition: 95.6%. This value is equivalent to the correct classification of 86 images out of a total of 90 photographs on the test dataset. In this sense, Table 5 presents the confusion matrix for the adoption of CNN8 + P (Repetition 1).

It can be seen in Table 5 that the CNN model correctly classified 43 images in the positive class and 43 images in the negative class (accuracy of 95.6%). Thus, for each class, CNN only missed 2 images in the test dataset (error around 4.44%). It is also worth noting that, in the study of [32], the maximum accuracy achieved was 90% for the same test images. Thus, indicating that the careful adjustment of the Data Augmentation hyperparameters can increase the results in the classification.

Table 6 summarizes the recommended hyperparameters for the analyzed database.

Table 7 Results of data augmentation hyperparameter tuning with logistic regression (second small dataset)

Hyperparameter	β	p	*Value*	x	OR
Rotation R. (R)	– 0.245	0.00	0	0	0.783
Hor. Flip (H)	0.071	0.07	False	0	1.074
Vertical Flip (V)	– 0.027	0.48	False	0	0.973
Height S. R. (He)	– 0.008	0.85	0	0	0.992
Shear Range (S)	0.089	0.02	0.2	1	**1.094**
Width S. R. (W)	– 0.024	0.53	0	0	0.976
Zoom Range (Z)	– 0.091	0.02	0	0	0.913

Bold value indicates the hyperparameters $OR > 1$ and $p < 0.05$

4.2 Results of second small dataset

This section report the results of applying the proposed methodology in a second small dataset: gutter integrity in roofs structures. In this sense, the Equation 11 presents the linear model adjusted for this case study:

$$\begin{aligned} y =& \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \cdot \\ & \cdot + \beta_4 x_4 + \beta_5 x_5 + \beta_6 x_6 + \beta_7 x_7 \\ =& 1.750 - 0.245 x_1 + 0.071 x_2 - 0.027 x_3 + \cdot \\ & \cdot - 0.008 x_4 + 0.089 x_5 - 0.024 x_6 - 0.091 x_7 \end{aligned} \tag{11}$$

Table 7 shows the results of the test statistic (p), recommended values and odds ratio (OR) per hyperparameter (second small dataset).

Table 7 shows that the only hyperparameter recommended by the HPtuningLogReg algorithm for the second case study was Shear Range (S), because $p < 0.05$ and $OR > 1$. On the other hand, four transformations did not reach statistical significance ($p > 0.05$): Horizontal Flip (H), Vertical Flip (V), Height Shift Range (He) and Width Shift Range (W). In addition, two hyperparameters achieved statistical effect ($p < 0.05$), but obtained $OR < 1$: Rotation Range (R) and Zoom Range (Z).

In this regard, the test stage was carried out with two Data Augmentation configurations:

- Proposed in this paper (P) - Hyperparameter Tuning for second small dataset: $S = 0.2$.
- Literature (L) - presented in [4] and used by [33]: $R = 40$; $H = TRUE$; $He = 0.2$; $S = 0.2$; $W = 0.2$; $Z = 0.2$.

Table 8 presents the accuracy results of test step to second small dataset.

Table 8 shows that the highest average accuracy (88.0%) for second case study was achieved by the CNN8 architecture with the proposed configuration. Moreover, this Data

Table 8 Maximum accuracy in the test step in each of the repetitions and respective mean accuracy (M) for second small dataset

Arch.	C.	1	2	3	4	5	M.
CNN8	P	83.3	**93.3**	**93.3**	86.7	83.3	**88.0**
CNN8	L	56.7	70.0	66.7	56.7	73.3	64.7
DenseNet	P	70.0	83.3	90.0	90.0	70.0	80.7
DenseNet	L	86.7	86.7	90.0	80.0	86.7	86.0
MobileNet	P	70.0	70.0	60.0	70.0	63.3	66.7
MobileNet	L	70.0	80.0	73.3	83.3	70.0	75.3

Bold values indicate the best result of test accuracy and mean accuracy

Comparison between methods recommended by the data augmentation configurations (Proposed (P) and Literature (L)) and architectures

Table 9 Comparison of this proposal with different papers that applied CNNs in the image processing of building construction: I [32], II [3], III [48] , IV [36] and V [54].

		Proposed	I [32]	II [3]	III [48]	IV [36]	V [54]
CNN application	Classification	✓	✓	–	–	–	–
	Detection	–	–	✓	✓	✓	–
	Segmentation	–	–	–	–	✓	✓
Problem	Crack detection	–	–	–	✓	✓	✓
	Bridge inspection	–	–	✓	–	–	–
	Roofs defects classification	✓	–	–	–	–	–
	Vegetation in facades	✓	✓	–	–	–	–
Analyzed Hyperparameters	Rotation Range	✓	✓	✓	✓	✓	✓
	Horinzontal Flip	✓	✓	✓	–	–	✓
	Vertical Flip	✓	–	–	–	–	✓
	Height Shift Range	✓	✓	–	–	–	–
	Shear Range	✓	✓	–	–	✓	–
	Width Shift Range	✓	✓	–	–	–	–
	Zoom Range	✓	✓	–	–	–	–
	Others	–	–	–	✓	✓	–
Tuning of Data Augmentation	Yes	✓	–	✓	✓	–	–
	No	–	✓	–	–	✓	✓

Augmentation combination ($S = 0.2$) achieved the maximum accuracy value in one iteration: 93.3%.

4.3 Comparison with other studies

In this section, a comparative study is carried out between the present proposal and other recent works in the literature: I [32], II [3], III [48], IV [36] and V [54]. For this, five features were observed: CNN application (classification, detection or segmentation), type of problem, analyzed hyperparameters and tuning of Data Augmentation methods. All analyzed paper applied Deep Learning models for image processing of building construction. Table 9 presents the comparison results.

From the Table 9 it is confirmed that the main contribution of this paper is the proposal of a methodology for tuning of data augmentation hyperparameters to building construction image classification, especially in vegetation recognition and roofs defects classification. In another way, it should be noted that most of the other studies in this area are dedicated to the problem of crack detection (or segmentation).

Furthermore, this proposal innovates by analyzing 128 (2^7) combinations of hyperparameters from seven Data Augmentation transformations: rotation range, horizontal flip, vertical flip, height shift range, shear range, width shift range and zoom range. In general, other papers analyze less combinations and transformations of Data Augmentation in the building construction image processing field.

Another important contribution is the proposal for the application of logistic regression models for hyperparameter tuning. The papers by [3] and [48] also present methodologies for recommending Data Augmentation hyperparameters. However, these studies are applied to other problems (crack detection or bridge inspection) and do not adopt logistic regression methods.

5 Conclusion

The objective of this paper was to propose a rigorous methodology for tuning of Data Augmentation hyperparameters in Deep Learning to small datasets. In this sense, the main contributions of this study are:

- Careful analysis of Data Augmentation transformations in the application of Deep Learning in building image classification, especially in the recognition of vegetation on facades and roofs defects classification.
- Design of experiments with 128 combinations of Data Augmentation using the Keras library and the R software.
- Proposal of the HPtuningLogReg method using Logistic Regression to tuning of Data Augmentation hyperparameters.
- Comparison of Data Augmentation configurations by adopting three Convolutional Neural Network architectures from the literature.

Regarding the results, in the first stage of experiments were recommended three Data Augmentation transformations for first case study: Height Shift Range (He), Width Shift Rang (W) and Zoom Range (Z). According to the Logistic Regression model, adopting $He = 0.2$ guarantees an increase of around 20% of success in the correct image classification. On the other hand, by adopting $W = 0.2$ or $Z = 0.2$ the chance of success is increased by approximately 12%. Moreover, from the second stage of experiments, the configuration ($He = 0.2$; $W = 0.2$; $Z = 0.2$) was the only one to present statistical significance ($p \leq 0.05$) for the three CNN architectures analyzed.

Finally, in the testing stage, the selected Data Augmentation configuration reached the highest average accuracy (92%) when adopting the CNN8 architecture. In addition, this combination also resulted in the greatest accuracy in one repetition: 95.6%. This value is equivalent to the correct classification of 86 images out of a total of 90 photographs on the test dataset of first case study.

For the second case study, the logistic regression model recommended the Shear Range transformation for Data Augmentation. In this sense, the hyperparameters selected for this application also achieved the best results in the test phase: 93.3%.

In future work, it is expected to analyze other Data Augmentation transformations. In addition, it is also suggested to test more levels for specific hyperparameters, for example, Zoom Range ranging from 10% to 50%. It is also worth highlighting the importance of investigating possible limitations of the logistic regression model, for example, the proposed approach did not accounted for the interactions among different predictor variables. With interaction effects, each predictor (hyperparameter) influences others and could result others solutions of tuning. Another important point will be the adoption of the HPtuningRegLog method in tuning of Data Augmentation settings in other applications with small dataset of building construction image classification.

Acknowledgements The authors are grateful to Research Group in Construction Technology and Management (GETEC) - School of Engineering (UFBA) - for providing the second image dataset, the Robotics & Perception Group (University of Zurich) for providing "The Zurich Urban Micro Aerial Vehicle Dataset", UFBA and UFRB.

Declarations

Conflict of interest The Authors listed in this article declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. Ali R, Cha Y-J (2019) Subsurface damage detection of a steel bridge using deep learning and uncooled micro-bolometer. Constr Build Mater 226:376–387
2. Barberousse H, Lombardo RJ, Tell G, Couté A (2006) Factors involved in the colonisation of building facades by algae and cyanobacteria in france. Biofouling 22(02):69–77
3. Bianchi E, Abbott AL, Tokekar P, Hebdon M (2021) Coco-bridge: Structural detail data set for bridge inspections. J Comput Civ Eng 35(3):04021003
4. Chollet F, Allaire JJ (2018) Deep learning with R. Manning Publications
5. Conceição J, Poça B, De Brito J, Flores-Colen I, Castelo A (2017) Inspection, diagnosis, and rehabilitation system for flat roofs. J Perform Constr Facil 31(6):04017100
6. Cubuk ED, Zoph B, Shlens J, Le QV (2020) Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp 702–703
7. da Silva GR, Valões DC, Nascimento CF, SNA A., Candeia MA, Santiago H, Oliveira DV, Everton G, Lima JC, Souza JM (2 021) Elaboration of a damage map the facades of a public building in the city of triunfo/pe. Int J Adv Eng Res Sci 8:237–244
8. Dung CV et al (2019) Autonomous concrete crack detection using deep fully convolutional neural network. Autom Constr 99:52–58
9. Elgendy M (2020) Deep learning for vision systems. Manning Publications
10. Fang W, Ding L, Luo H, Love PE (2018) Falls from heights: a computer vision-based approach for safety harness detection. Autom Constr 91:53–61
11. Gao Y, Mosalam KM (2018) Deep transfer learning for image-based structural damage recognition. Comput-Aid Civ Infrastruct Eng 33(9):748–768
12. Garcez N, Lopes N, de Brito J, Silvestre J (2012) System of inspection, diagnosis and repair of external claddings of pitched roofs. Constr Build Mater 35:1034–1044

13. Giolo S. R (2017). *Introduction to categorical data analysis with applications (in portuguese)*. Editora Blucher
14. Guo J, Wang Q, Li Y, Liu P. (2020). Façade defects classification from imbalanced dataset using meta learning-based convolutional neural network. *Computer-Aided Civil and Infrastructure Engineering*
15. He K, Zhang X, Ren S, Sun J (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778
16. Hosmer DW Jr, Lemeshow S, Sturdivant RX (2013) Applied logistic regression, vol 398. John Wiley & Sons
17. Howard A. G, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861
18. Huang G, Liu Z, Van Der Maaten L, Weinberger K. Q (2017). Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269
19. Hutter F, Hoos H, Leyton-Brown K. (2014). An efficient approach for assessing hyperparameter importance. In *Proceedings of International Conference on Machine Learning 2014 (ICML 2014)*, pages 754–762
20. Hutter F, Kotthoff L, Vanschoren J, editors (2019). *Automated Machine Learning: Methods, Systems, Challenges*. Springer. In press, available at http://automl.org/book
21. Kaamin M, Ahmad N, Razali S, Mokhtar M, Ngadiman N, Masri D, Hussin I, Asri L. (2020). 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. volume 1529
22. Kolar Z, Chen H, Luo X (2018) Transfer learning and deep convolutional neural networks for safety guardrail detection in 2d images. Autom Constr 89:58–70
23. Lecun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
24. Loukma M, Stefanidou M (2018) Causes of deterioration of ottoman mosques. WIT Transactions on The Built Environment 177:173–180
25. Maeda H, Sekimoto Y, Seto T, Kashiyama T, Omata H (2018) Road damage detection and classification using deep neural networks with smartphone images. Computer-Aided Civil and Infrastructure Engineering 33(12):1127–1141
26. Majdik AL, Till C, Scaramuzza D (2017) The zurich urban micro aerial vehicle dataset. The International Journal of Robotics Research 36(3):269–273
27. Mantovani RG, Rossi AL, Alcobaça E, Vanschoren J, de Carvalho AC (2019) A meta-learning recommender system for hyperparameter tuning: Predicting when tuning improves svm classifiers. Inf Sci 501:193–221
28. Monshi MMA, Poon J, Chung V, Monshi FM (2021) Covidxraynet: Optimizing data augmentation and cnn hyperparameters for improved covid-19 detection from cxr. Computers in Biology and Medicine 133:104375
29. Myers RH, Montgomery DC, Anderson-Cook CM (2016) Response surface methodology: process and product optimization using designed experiments. John Wiley & Sons
30. Neary P. (2018). Automatic hyperparameter tuning in deep convolutional neural networks using asynchronous reinforcement learning. In *2018 IEEE International Conference on Cognitive Computing (ICCC)*, pages 73–77
31. Ottoni ALC, Nepomuceno EG, de Oliveira MS, de Oliveira DCR (2020) Tuning of reinforcement learning parameters applied to sop using the scott-knott method. Soft Comput 24:4441–4453
32. Ottoni ALC, Novo MS (2021) A deep learning approach to vegetation images recognition in buildings: a hyperparameter tuning case study. IEEE Lat Am Trans 19(12):2062–2070
33. Ottoni A. L. C, Novo M. S, Costa D. B. (2021). Hyperparameter tuning of convolutional neural networks for building construction image classication. *The Visual Computer*
34. Pawara P, Okafor E, Schomaker L, Wiering M. (2017). Data augmentation for plant classification. In *International Conference on Advanced Concepts for Intelligent Vision Systems*, pages 615–626. Springer
35. R Core Team (2020) R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria
36. Ren Y, Huang J, Hong Z, Lu W, Yin J, Zou L, Shen X (2020) Image-based concrete crack detection in tunnels using deep fully convolutional networks. Construction and Building Materials 234:117367
37. Rocha E, Macedo J, Correia P, Monteiro E (2018) Adaptation of a damage map to historical buildings with pathological problems: Case study at the church of carmo in olinda, pernambuco. Revista ALCONPAT 8(1):51–63
38. Sajedi SO, Liang X (2021) Uncertainty-assisted deep vision structural health monitoring. Computer-Aided Civil and Infrastructure Engineering 36(2):126–142
39. Schratz P, Muenchow J, Iturritxa E, Richter J, Brenning A (2019) Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. Ecol Model 406:109–120
40. Shankar K, Zhang Y, Liu Y, Wu L, Chen C-H (2020) Hyperparameter tuning deep learning for diabetic retinopathy fundus image classification. IEEE Access 8:118164–118173
41. Shen J, Xiong X, Li Y, He W, Li P, Zheng X (2021) Detecting safety helmet wearing on construction sites with bounding-box regression and deep transfer learning. Computer-Aided Civil and Infrastructure Engineering 36(2):180–196
42. Shorten C, Khoshgoftaar T. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1). cited By 456
43. Silveira, B., Melo, R., and Costa, D. B. (2021). Using uas for roofs structure inspections at post-occupational residential buildings. In Toledo Santos, E. and Scheer, S., editors, *Proceedings of the 18th International Conference on Computing in Civil and Building Engineering*, pages 1055–1068, Cham. Springer International Publishing
44. Song C, Xu W, Wang Z, Yu S, Zeng P, Ju Z. (2020). Analysis on the impact of data augmentation on target recognition for uav-based transmission line inspection. *Complexity*, 2020
45. Staffa L. B, Sa L. S. V, Lima M. I. S. C, Costa D. B. (2020). Use of image processing techniques for inspection of building roof structures for technical assistance purposes (in portuguese). *ENTAC - National Meeting of the Built Environment Technology*
46. Wang J.-J, Liu Y.-F, Nie X, Mo Y. (2022). Deep convolutional neural networks for semantic segmentation of cracks. *Structural Control and Health Monitoring*, 29(1). cited By 0
47. Wang X, Zhao Y, Pourpanah F (2020) Recent advances in deep learning. Int J Mach Learn Cybern 11:747–750
48. Wang Z, Yang J, Jiang H, Fan X (2020) Cnn training with twenty samples for crack detection via data augmentation. Sensors 20(17):4849
49. Xue Y, Li Y (2018) A fast detection method via region-based fully convolutional neural networks for shield tunnel lining defects. Computer-Aided Civil and Infrastructure Engineering 33(8):638–654
50. Yang Z, He B, Liu Y, Wang D, Zhu G (2021) Classification of rock fragments produced by tunnel boring machine using convolutional neural networks. Automation in Construction 125:103612

51. Younis MC, Keedwell E (2019) Semantic segmentation on small datasets of satellite images using convolutional neural networks. Journal of Applied Remote Sensing 13(4):046510
52. Zeng S, Zhang B, Zhang Y, Gou J (2020) Dual sparse learning via data augmentation for robust facial image classification. Int J Mach Learn Cybern 11(8):1717–1734
53. Zhou S, Song W. (2020). Deep learning-based roadway crack classification using laser-scanned range images: A comparative study on hyperparameter selection. *Automation in Construction*, 114
54. Zhou S, Song W (2021) Crack segmentation through deep convolutional neural networks and heterogeneous image fusion. Automation in Construction 125:103605