






Cracks Detection in Concrete Surfaces using Machine Learning

Detecção de Fissuras em Superfícies de Concreto utilizando Machine Learning

Antonio Mendes Magalhães Júnior¹ , Ícaro Viterbre Debique Sousa² , Heron Viterbre Debique Sousa³ , Thelma Sáfiadi⁴ , Paulo Henrique Sales Guimarães⁵ 

Received: September 11, 2024

Received in revised form: December 17, 2024

Accepted: February 18, 2025

Available online: April 8, 2025

ABSTRACT

In this study, Machine Learning techniques were used to detect cracks in concrete surfaces. Texture descriptors were extracted from a set of images of cracked and non-cracked concrete surfaces using Gray Level Co-occurrence Matrix (GLCM). Texture descriptors features were used to train four machine learning models: Logistic Regression, Artificial Neural Networks, Random Forest, and eXtreme Gradient Boosting (XGBoost). k -fold cross validation was used to evaluate the performance of the models as well as metrics such as accuracy, sensitivity, precision, F1-score - the harmonic mean between precision and sensitivity - and the Area Under the ROC Curve (AUC). The results indicate satisfactory performance of all models, with emphasis on the XGBoost model, which achieved 99.65% accuracy and 99.70% sensitivity.

keywords infrastructure maintenance, computer vision, texture descriptors, automated monitoring

RESUMO

Neste estudo foram empregadas técnicas de Aprendizado de Máquina para a detecção de fissuras em superfícies de concreto. Foi utilizado um conjunto de imagens de superfícies de concreto com e sem fissuras. A partir dessas imagens, foram extraídos descritores de textura por meio de matrizes de coocorrência de níveis de cinza (GLCM). Utilizando os descritores de textura como características, foram treinados quatro modelos de aprendizado de máquina, sendo: Regressão Logística, Redes Neurais Artificiais, Random Forest e eXtreme Gradient Boosting (XGBoost). A validação cruzada do tipo k -fold foi utilizada para avaliar o desempenho dos modelos, além de métricas como acurácia, sensibilidade, precisão, F1-score – média harmônica entre precisão e sensibilidade – e a Área Sob a Curva ROC (AUC). Os resultados mostraram desempenho satisfatório de todos os modelos, com destaque para o modelo XGBoost, que atingiu 99,65% de acurácia e 99,70% de sensibilidade.

palavras-chave manutenção de infraestrutura, visão computacional, descritores de textura, monitoramento automatizado

¹PhD student in statistics at UFLA., UFLA, Lavras, Minas Gerais, Brazil. amendesmj@gmail.com

²PhD student in statistics at UFLA, UFLA, Lavras, Minas Gerais, Brazil. viterbre@gmail.com

³PhD student in Metallurgical Engineering, UFMG, Belo Horizonte, Minas Gerais, Brazil. heronviterbre@gmail.com

⁴Prof.Dr., Department of Statistics, UFLA, Lavras, Minas Gerais, Brazil. safadi@ufla.br

⁵Prof.Dr., Department of Statistics, UFLA, Lavras, Minas Gerais, Brazil. paulo.guimaraes@ufla.br

Introduction

Reinforced concrete is widely used throughout the world and is essential for construction projects such as bridges, viaducts, and buildings. This type of concrete is constituted by concrete and steel reinforcement, which results in a robust and durable structure. However, cracks in the concrete can allow the entry of pathogenic agents, such as moisture, chlorides, and carbon dioxide. The entry of these agents accelerates the reinforcement corrosion and the concrete degradation, compromising the integrity of the structure (Arivabene, 2015; Nascimento & Fontes, 2021).

Crack detection is traditionally performed by visual inspections, but this method faces some challenges. Cracks can be very small, which makes them difficult to detect, especially in the early stages. Dirt on the concrete surface and inadequate lighting are other factors that can also disrupt detection. In addition, the need to access hard-to-reach areas may require specialized equipment and increase the inspection risk.

Although there are standards for the classification and tolerance of cracks, such as NBR 9575 (Associação Brasileira de Normas Técnicas [ABNT], 2010) and NBR 6118 (Associação Brasileira de Normas Técnicas [ABNT], 2014) visual inspections contribute to the subjectivity issue. Different inspectors may have different perceptions about the severity and extent of these cracks. Therefore, the early and accurate identification of these cracks is of great importance to ensure the safety of the buildings.

Advances in computing power have enabled the application of machine learning techniques in several areas, including civil engineering. Considering the identification of cracks and other damage in concrete structures, several studies have explored methodologies based on computer vision. Kim et al. (2020) suggested the use of convolutional neural networks (CNNs) for crack detection and segmentation. The study achieved a final accuracy of 99.98%, demonstrating high effectiveness in the classification task.

Diniz et al. (2023) approached not only cracks but also other structural pathologies, such as exposed steel bars and efflorescence. Their methodology combined the detection of pathological areas with the YOLOv4 model in wide images and the classification of specific pathologies with CNNs in cropped images. For the crack classification task, the model achieved an accuracy of 99.4%. Chen et al. (2019) focused on optimized convolutional networks, applying techniques such as Adam and Batch Normalization to increase the efficiency of the model in detecting cracks in segmented images. The developed model achieved an accuracy of 99.71%, confirming the effectiveness of the approach for the automated monitoring of cracks in concrete structures.

Although these studies have achieved significant results, they use deep neural networks and more complex processing steps, which can result in high computational costs and require more powerful machines to train and implement the models. This proposal differs by using texture descriptors, extracted through gray-level co-occurrence matrices (GLCM), as predictors for machine learning models. This approach allows the construction of simpler models, such as Multilayer Perceptron (MLP) artificial neural networks, which require lower computational costs. These models can be trained and used on conventional computers, without compromising the accuracy in crack detection. In addition, this study suggests the use of four different models to validate the effectiveness of the proposed strategy.

Thus, the present study aimed to investigate the application of machine learning in the detection of cracks in concrete surfaces. To do that, different machine learning models were trained and evaluated. Metrics such as accuracy, Kappa index, AUC, sensitivity, specificity and precision were used to verify the performance of the models. *k*-fold cross-validation was used to ensure greater reliability of the results obtained.

Methodology

Using a set of concrete images with and without cracks, the Haralick texture descriptors were extracted through Gray Level Co-Occurrence Matrices. Then, four machine learning models, Logistic Regression, Multilayer Perceptron (MLP) Artificial Neural Networks Random Forest and XGBoost were developed and evaluated. Each one of these steps is detailed below.

Data preprocessing

A set of concrete surfaces images obtained from Özgenel (2019) was used. This set contains 20,000 images with a resolution of 227×227 pixels, divided equally between classes with and without cracks.

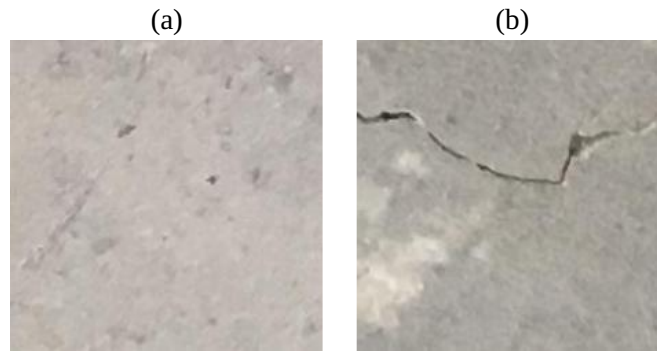
The dataset was generated from 458 high-resolution images (4032×3024 pixels), presenting variations in terms of superficial finish and lighting conditions. The images were collected from several buildings on the campus of the Middle East Technical University (METU) in Ankara, Turkey.

Initially, all collected images were in RGB (Red, Green and Blue) color format. For texture analysis using Haralick descriptors, it was necessary to convert these images to the grayscale. This process was performed using the R software and the *wvtool* package. The conversion used the weighted formula that considers different contributions from the red, green and blue colors, given by equation (1):

$$\text{Gray} = 0.30 \cdot R + 0.59 \cdot G + 0.11 \cdot B. \quad (1)$$

Examples of surfaces with and without cracks are shown in Figure 1.

Figure 1 - Illustrates surfaces with and without cracks, (a) and (b) respectively.



From adapted “Concrete crack images for classification”, by Ç. F. Özgenel, 2019, Mendeley Data, 2.

Texture descriptors

Texture descriptors were extracted with the objective of composing the data that served as predictors in the machine learning models. To extract these descriptors, Gray Level Co-Occurrence Matrix (GLCM) was used, which is constructed according to the frequency with which different combinations of gray levels occur in an image, considering a displacement distance (d) and an angular direction (θ). That way, information about the spatial distribution of the gray levels in the image is captured.

The GLCMs were calculated for each image considering four angular directions (0° , 45° , 90° and 135°) and displacement distance $d = 1$, in other words, the immediate neighboring pixel. For each metric, the average of the four angular directions was used to compose the data. Using the matrices, all texture descriptors described in Haralick et al. (1973) were extracted. This process was performed using the R software, through the functions available in the *wvtools* package.

Among Haralick’s texture descriptors, the following ones excel in the literature: homogeneity, which assesses the uniformity of gray levels; contrast, which measures the variation in intensity between a pixel and its neighbors; correlation, which assesses the correlation between pixels along a given direction; entropy, which refers to the degree of disorder in the image; and the inverse difference moment, or local homogeneity, which measures the local uniformity of gray levels in the image (Mansour & Thomson, 2023; Tou et al., 2007; Vrbik et al., 2019).

Some of the Haralick descriptors are highly correlated, which can be problematic for models that are more sensitive to multicollinearity (Pacifi et al., 2009). Multicollinearity occurs when two or more predictors are highly correlated and can lead to erroneous interpretations of model results. In models that are more sensitive to multicollinearity, Principal Component Analysis (PCA) was adopted as a strategy to minimize this problem, transforming the descriptors into a set of components that are not correlated with each other.

Adjusted models

Using the *caret* package in the R software (R Core Team, 2024), four distinct models were developed for crack detection: Logistic Regression, Artificial Neural Networks, Random Forest, and XGBoost. The data

were divided in a stratified manner, with 80% destined for training and 20% for testing. This proportion is widely adopted in the literature, as it balances the quantity of elements necessary to adjust the models with the portion intended to an independent evaluation. Thus, two sets were obtained: the training set, used to adjust and validate the models, and the test set, used to evaluate the final performance independently.

The Logistic Regression model was configured by varying the parameters α and λ , i.e., $0.10 \leq \alpha \leq 1$ and $0.0008 \leq \lambda \leq 0.0800$. These parameters are responsible for adjusting the regularization penalty in the model, where α controls the balance between L1 and L2 regularizations, and λ determines the intensity of the penalty. Centralization, scaling, and PCA were used as pre-processing, due to the sensitivity of this model to the problem of multicollinearity.

The Random Forest was configured with different numbers of sampled variables in each division (2, 8 and 15). This parameter is responsible for determining the number of predictors considered in each node division. A greater number of sampled variables can increase the robustness of the model, but it also increases the processing time. Centralization and scaling were used as preprocessing.

At last, XGBoost was configured by varying the learning rate (0.3 and 0.4), which controls the speed at which the model fits the data during training; the number of iterations (50, 100, and 150), which is responsible for the number of times the model traverses the data to optimize the parameters; the maximum depth of the trees (1, 2, and 3), which determines the complexity of the individual trees; and the fractions of predictors sampled from each tree (0.6 and 0.8) and samples used to train each tree (0.50, 0.75, and 1.00), which are responsible for regulating the diversity and robustness of the model by controlling the proportion of variables and samples used in each tree, respectively. The preprocessing used included centering and scaling.

To adjust the hyperparameters of each model, a grid search was used to test all possible combinations within the set intervals. Using this approach ensures an exhaustive evaluation of the hyperparameters, identifying the configurations that maximize the performance of the models. The best values found for each model were applied in the final evaluation phase.

The k -fold validation with $k = 10$ was used to evaluate the performance of the models. In this procedure, the training set was divided into ten parts. In each of the ten iterations, nine parts were used for training and one part was used for validation. The iterations ensure that each part is used as validation exactly once. This procedure was repeated five times, each time with different partitioning of the training set, minimizing the variability of the results due to randomness in the initial division of the data. The final configuration of each model was defined based on the hyperparameters that resulted in the best average performance over the five repetitions, considering accuracy as the main metric.

The models were also evaluated based on several performance metrics calculated from their responses in the test data set, such as accuracy, Kappa Index, Area Under the ROC Curve (AUC), sensitivity, specificity, precision, balanced accuracy and F1-score.

Accuracy measures the proportion of correct classifications, while the Kappa index evaluates the concordance between the model classifications and the real ones, in addition to the expectation by fortuity. AUC determines the ability of the model to distinguish between classes, the sensitivity measures the correct identification of cracks, and specificity evaluates the correct identification of surfaces without cracks. Precision calculates the proportion of true positives among all positive predictions, and balanced accuracy is the average of sensitivity and specificity. The $F1$ -score metric, being the harmonic mean between precision and sensitivity, provides a balance between the two. This set of metrics allowed us to evaluate the performance of the models, considering both the crack detection and the prevention of false positives.

Results

The training and evaluation of the models allowed an analysis of their performance in detecting cracks in concrete surfaces. The results presented below allow comparison between the trained models.

The final models of each type were selected based on their accuracy. For Logistic Regression, the final model presented values of $\alpha = 1$ and $\lambda = 0.0008001586$. The ANN model was configured with 15 neurons in the hidden layer. Regarding the Random Forest, the final configuration included 8 variables sampled in each division. The XGBoost model was adjusted with a learning rate of 0.4, 150 iterations, maximum tree depth of 3, fraction of predictors sampled in each tree of 0.8 and 1.0 samples used to train each tree.

The performance metrics of the models were calculated during training using k -fold cross-validation (with

$k = 10$), before the final evaluation on the test set. Figure 2 shows the accuracy and Kappa index boxplots, based on the values obtained in each iteration of the cross-validation procedure performed on the training set, allowing an analysis of the consistency of the models' performance.

Figure 2 - Boxplots of accuracy and Kappa Index.

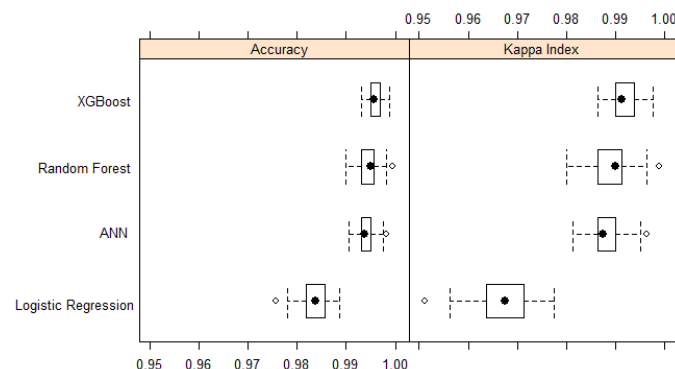
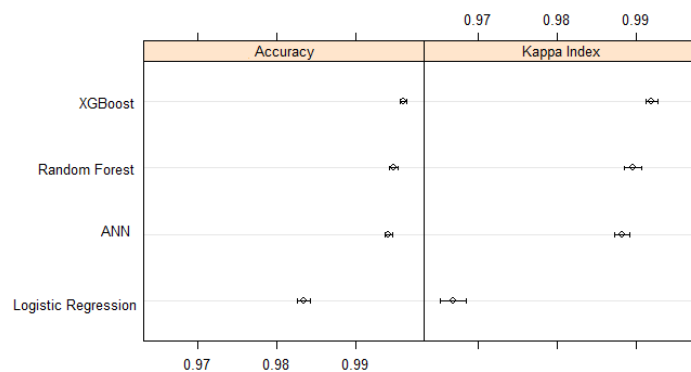


Figure 3 presents the 95% confidence intervals for these metrics, allowing a statistical evaluation of the variability and stability of the models during the training stage.

Figure 3 - Accuracy confidence intervals and Kappa Index.



It is possible to observe that, during cross-validation in the training stage, the XGBoost model presented the highest values for accuracy and Kappa Index, indicating superior performance compared to the other models. In the boxplots, no outliers were observed, suggesting consistent performance throughout the iterations. The 95% confidence intervals for both metrics were narrow and did not overlap with those of the other models, reinforcing the statistically significant superiority of XGBoost.

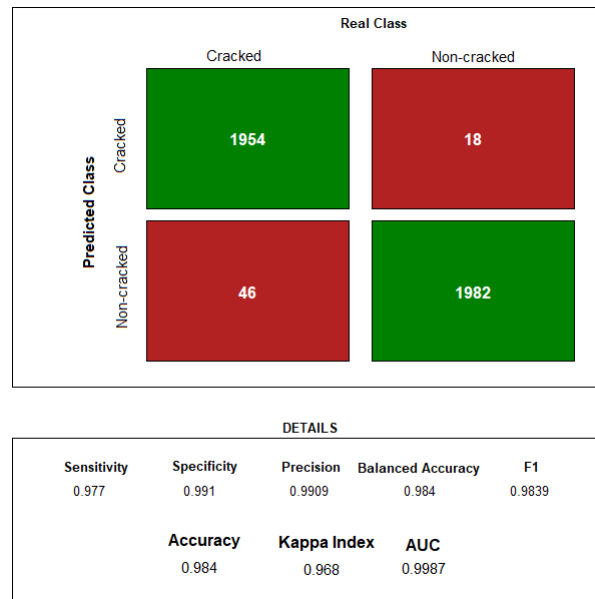
The Random Forest and ANN models presented similar performances, with slightly lower accuracy and Kappa Index than XGBoost. Both presented outliers in the boxplots, reflecting lower consistency in some iterations. There is overlap in their 95% confidence intervals, indicating that there was no statistically significant difference between the average performances of these models, considering accuracy and Kappa Index, during this stage.

The Logistic Regression, despite its simplicity, demonstrated reasonable performance, but presented greater dispersion in accuracy and Kappa Index values compared to the other models. The 95% confidence intervals were wider, especially for the Kappa Index, indicating greater instability in the results. Furthermore, there was no overlap with the intervals of the other models, confirming that Logistic Regression had significantly lower performance.

For the independent test data, that is, the 20% of the data reserved exclusively for final evaluation and that were not used in any stage of training or validation, confusion matrices were constructed to evaluate the performance of each model. From these matrices, the accuracy metrics, Kappa Index, AUC, sensitivity, specificity, precision, balanced accuracy and F1-score were calculated. The 95% confidence interval for accuracy was estimated based on the proportion of correct answers obtained by the models and the total number of samples present in the test data, using the binomial distribution method.

Figure 4 shows the confusion matrix and performance metrics for the Logistic Regression model.

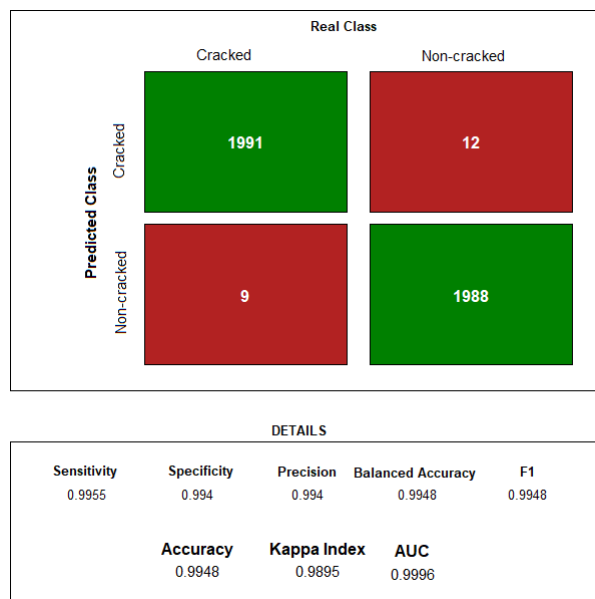
Figure 4 - Confusion matrix and evaluation metrics of the Logistic Regression model.



The Logistic Regression model performed well in the crack detection task, with an accuracy of 98.4% (95% CI: 97.96% – 98.77%) and a Kappa Index of 0.9680. The confusion matrix indicates 1954 true positives and 1982 true negatives, with only 46 false negatives and 18 false positives. The sensitivity was 0.9770 and the specificity was 0.9910, highlighting the model's ability to correctly identify both cracked and non-cracked surfaces. The accuracy test compared to No Information Rate (NIR), which compares the accuracy of the classification model with the accuracy expected by a random classifier, resulted in a *p*-value close to zero, reinforcing the superiority of the model in relation to a random classifier.

Figure 5 shows the confusion matrix and performance metrics for the Artificial Neural Network model.

Figure 5 - Confusion matrix and evaluation metrics of the Artificial Neural Network model.

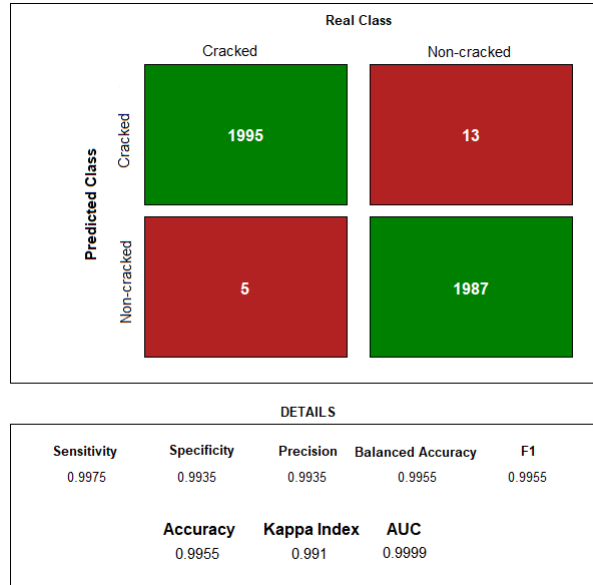


The Artificial Neural Networks model presented high performance in crack detection, with an accuracy of 99.48% (95% CI: 99.20% – 99.67%) and a Kappa Index of 0.9895. The confusion matrix indicates 1991 true positives and 1988 true negatives, with only 9 false negatives and 12 false positives. The sensitivity was

0.9955 and the specificity was 0.9940, highlighting the model's ability to correctly identify both cracked and uncracked surfaces. The accuracy test compared to NIR resulted in a p -value close to zero, reinforcing the superiority of the model in relation to a random classifier.

Figure 6 shows the confusion matrix and performance metrics for the Random Forest model.

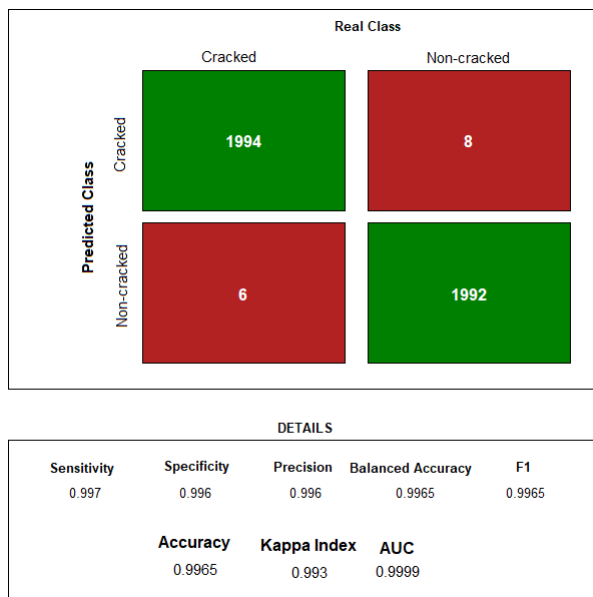
Figure 6 - Confusion matrix and evaluation metrics of the Random Forest model.



The Random Forest model also performed very well, with an accuracy of 99.55% (95% CI: 99.29% – 99.73%) and a Kappa index of 0.9910. The confusion matrix indicates 1995 true positives and 1987 true negatives, with only 5 false negatives and 13 false positives. The sensitivity was 0.9975 and the specificity was 0.9935, which shows that the model can correctly classify both cracked and uncracked surfaces. The accuracy test compared to NIR resulted in a p -value close to zero, which shows that the model is significantly better than a random classifier.

At last, Figure 7 shows the confusion matrix and performance metrics for the XGBoost model.

Figure 7 - Confusion matrix and evaluation metrics of the XGBoost model.



The XGBoost model showed excellent performance in crack detection, with an accuracy of 99.65% (95% CI: 99.41% – 99.81%) and a Kappa Index of 0.9930. The confusion matrix indicates 1994 true

positives and 1992 true negatives, with only 6 false negatives and 8 false positives. The sensitivity was 0.9970 and the specificity was 0.9960. The accuracy test compared to NIR resulted in a p-value close to zero, indicating that the performance of the classifier is also significantly superior to that of a random classifier.

Conclusions

The comparative analysis of the four machine learning models showed that all models performed well in the task of detecting cracks in concrete surfaces. The Logistic Regression model achieved an accuracy of 98.4% (95% CI: 97.96% – 98.77%) and a Kappa Index of 0.9680. The Artificial Neural Networks model presented an accuracy of 99.48% (95% CI: 99.20% – 99.67%) and a Kappa Index of 0.9895. The Random Forest had a similar performance, with an accuracy of 99.55% (95% CI: 99.29% – 99.73%) and a Kappa Index of 0.9910. At last, the XGBoost stood out as the superior model, with an accuracy of 99.65% (95% CI: 99.41% – 99.81%) and a Kappa Index of 0.9930, showing excellent performance in the task.

By using a simpler and less computational cost methodology, based on the extraction of texture descriptors through gray-level co-occurrence matrices, this study demonstrates the potential to expand the use of machine learning in the detection of cracks in concrete surfaces. This approach facilitates the application in different scenarios by allowing the use of a wider range of machine learning models from the extracted texture descriptors, making the process more accessible, flexible and efficient compared to techniques that rely on deep neural networks.

The results of this study indicate that the proposed models can be effectively applied to improve the crack detection process in concrete surfaces. The adoption of these models as an aid to the inspection process could reduce the need for on-site visual inspections, increase detection accuracy, and improve preventive maintenance of infrastructures.

Future papers could explore the detection of other pathologies in concrete structures, such as corrosion, disintegration and efflorescence. Machine learning models could be developed to automatically identify and classify these pathologies and their severity levels. These classifiers could be integrated into monitoring and automatic alert systems, increasing the efficiency of infrastructures preventive maintenance.

Acknowledgements

The authors would like to thank the Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) for funding this study. We would also like to thank the journal's reviewers for their contributions and suggestions, which helped to improve this paper.

CRedit authorship contribution statement

A. M. Magalhães Júnior contributed to the conception, data curation, investigation, methodology, formal analysis, writing, and editing. I. V. Debique Sousa was involved in the conception, investigation, formal analysis, writing, and editing. H. V. D. Sousa participated in the conception and writing. T. Sáfadi and P. H. S. Guimarães contributed to supervision and revision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

Data availability

Aiming to contribute to the reproducibility of the experiments, the code developed for data processing, model training and analysis of the results presented in this study is available upon request to the authors.

References

- Arivabene, A. C. (2015). Patologias em estruturas de concreto armado: Estudo de caso. *Revista Especialize On-line IPOG*, 1(10), 1–22.

- Associação Brasileira de Normas Técnicas. (2010). *NBR 9575: Impermeabilização - Seleção e projeto*. ABNT.
- Associação Brasileira de Normas Técnicas. (2014). *NBR 6118: Projeto de estruturas de concreto - Procedimento*. ABNT.
- Chen, K., Yadav, A., Khan, A., Meng, Y., & Zhu, K. (2019). Improved crack detection and recognition based on convolutional neural network. *Modelling and Simulation in Engineering, 2019*, 1–8. <https://doi.org/10.1155/2019/8796743>
- Diniz, J. d. C. N., Paiva, A. C., Braz, J., G., Almeida, J. D. S., Silva, A. C., Cunha, A. M. T. S., & Cunha, S. C. A. P. S. (2023). A method for detecting pathologies in concrete structures using deep neural networks. *Applied Sciences, 13*(9), 5763. <https://doi.org/10.3390/app13095763>
- Haralick, R. M., Shanmugam, K., & Dinstein, I. H. (1973). Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics, 6*(6), 610–621. <https://doi.org/10.1109/TSMC.1973.4309314>
- Kim, J. J., Kim, A.-R., & Lee, S.-W. (2020). Artificial neural network-based automated crack detection and analysis for the inspection of concrete structures. *Applied Sciences, 10*(22), 8105. <https://doi.org/10.3390/app10228105>
- Mansour, I. R., & Thomson, R. M. (2023). Haralick texture feature analysis for characterization of specific energy and absorbed dose distributions across cellular to patient length scales. *Physics in Medicine & Biology, 68*(9), 1–12. <https://doi.org/10.1088/1361-6560/acc7f1>
- Nascimento, E. R. d. S., & Fontes, M. D. S. (2021). Patologias das estruturas de concreto armado. *Revista FATEC de Tecnologia e Ciências, 6*(1), 1–28. <https://revista.fateciba.edu.br/index.php/rftc/issue/view/rftcv6n12021>
- Özgenel, Ç. F. (2019). Concrete crack images for classification. *Mendeley Data, 2*. <https://doi.org/10.17632/5y9wdsg2zt.2>
- Pacifici, F., Chini, M., & Emery, W. J. (2009). A neural network approach using multiscale textural metrics from very high-resolution panchromatic imagery for urban land use classification. *Remote Sensing of Environment, 113*(6), 1276–1292. <https://doi.org/10.1016/j.rse.2009.02.014>
- R Core Team. (2024). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing.
- Tou, J., Tou, P., Lau, P. Y., & Tay, Y. H. (2007). Computer vision-based wood recognition system. In Thai Embedded Systems Association, *Proceedings of the International Workshop on Advanced Image Technology* [Proceedings]. International Workshop on Advance Image Technology, Bangkok, Thailand.
- Vrbik, I., Van Nest, S. J., Meksiarun, P., Loeppky, J., Brolo, A., Lum, J. J., & Jirasek, A. (2019). Haralick texture feature analysis for quantifying radiation response heterogeneity in murine models observed using raman spectroscopic mapping. *PLoS One, 14*(2), e0212225. <https://doi.org/10.1371/journal.pone.0212225>