IT Journal Research and Development (ITJRD)
Vol.10, No.2, March 2026, E-ISSN : 2528-4053 | P-ISSN : 2528-4061
DOI : 10.25299/itjrd.2025.20560

Comparative Analysis of SVM and XGBoost Classifiers with HOG Features for Concrete Crack Detection

Ridha Adjie Eryadi
Department of Informatics, Digital Technology University
ridhaadjie@digitechuniversity.ac.id

Article Info

Article history:

Received Dec 12, 2024
Revised Mar 14, 2025
Accepted Sep 18, 2025

Keyword:

Comparative Analysis
Concrete Crack Detection
HOG
Support Vector Machine
XGBoost Classifier

ABSTRACT

This study offers a comparative assessment of the Support Vector Machine with Radial Basis Function Kernel and Extreme Gradient Boosting for automated concrete crack detection based on Histogram of Oriented Gradients feature extraction. Data comprised 40,000 RGB concrete images from an open-source Mendeley dataset; half were cracked and half were non-cracked. They processed through a preprocessing pipeline that includes the Poisson noise reduction and bilateral filtering techniques. Two approaches, holdout validation over several training/testing configurations (50:50, 60:40, 70:30, and 80:20) and systematic 5-fold cross-validation, were adopted for evaluation of the Wilcoxon signed-rank test for statistical significance and inference time for computational efficiency assessment. The experimental results indicate that SVM achieved a better holdout accuracy of 98.94% with the 80:20 configuration, while XGBoost achieved a cross-validation mean accuracy of 98.83% ± 0.0015. However, no statistically significant performance difference was revealed between the models according to the Wilcoxon analysis. Results indicated SVM excels at minimising false positives on undamaged surfaces, whereas XGBoost is better for identifying cracks, meaning that the choice of models used should depend on the application requirements, where applications require either the minimisation of false alarms or maximum sensitivity for detection in the case of structural health monitoring.

Corresponding Author:

Ridha Adjie Eryadi
Departement of Informatics
Digital Technology University
Jalan Cibogo Indah III, Bodogol, Kota Bandung, Indonesia
Email: ridhaadjie@digitechuniversity.ac.id

1. INTRODUCTION

Geographically, Indonesia is located in the Pacific Ring of Fire, which makes it particularly vulnerable to earthquakes, so Indonesia ranks as one of the most vulnerable countries in the world regarding earthquakes. The country is located on active tectonic plates, so it receives earthquakes frequently and violently [1]. The analysis of earthquake-related losses reveals that most death incidences occur because of the collapse or tearing apart of structures rather than earth-shaking [2]. This kind of geological vulnerability indicates the major need for improved infrastructure related to public safety in any regional development. Indonesia has initiated an integrated infrastructure development plan for all these aspects, from roads to bridges, railways, and ports. It works towards improving connectivity and mobility and building structural resilience for possible earthquakes [3].

The detection of structural cracks plays an essential role in the safety operation of infrastructure. Cracks on structures are critical in predicting deterioration and condition, calling for proactive time frames for remedial interventions [4]. Several methodologies for crack detection exist, from time to time, such as visual inspections, specialized measurement instrumentation, and examinations conducted by experts; however, these classical ones are mostly done manually by survey protocols. Such conventional reconnaissance methods are inherently resource and productivity-consuming. The manual inspection process is cost-intensive and labor-dependent, which opens room for subjective time-bound constraints that will influence assessment accuracy [5, 6].

Conventional visual inspections had built-in limitations that drove the need to advance better, more efficient, and standardized modes of infrastructure assessment and monitoring [7]. One of the most critical problems in structural engineering is accurately identifying and classifying safety-critical cracks among many structural imperfections. The complexity of this task has impelled research initiatives toward implementing intelligent crack identification systems underpinned by computer vision technologies [8]. These high-end computational approaches have been revolutionary tools in infrastructure inspection without comparison in defect detection and analysis. They've affected computer vision-based methods for different applications in infrastructure evaluation and are quite different from the conventional practices of structural health monitoring [7].

Computer vision-based detection systems generally have a two-step approach in which an image dataset is first subjected to feature extraction using some edge detection algorithms such as Sobel, Canny, and Prewitt filters. This is followed by advanced feature descriptors, including Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT), and Speeded-Up Robust Features (SURF) [9,10]. These feature vectors are usually the input to a variety of classification algorithms, including the Support Vector Machine (SVM), k-nearest Neighbors (KNN), Naive Bayes and ensemble methods (e.g., Adaptive Boosting (AdaBoost) or eXtreme Gradient Boosting (XGBoost) [11,12,13], or deep learning approaches Convolutional Neural Networks (CNN) [2,5,6].

This research aims to fill the gap by bridging the absence of previous comparisons of high-performing classical machine learning methods under a unified pipeline for preprocessing and feature extraction. While the SVM model has consistently proven high generalization ability in high-dimensional feature spaces [14,15], while also being resilient toward overfitting on smaller datasets, XGBoost has also been touted as one of the most accurate and scalable ensemble methods [16-18], which excels in noisy features and imbalanced datasets. Yet, not much has been done to compare these two methods on the same crack detection dataset with respect to preprocessing and feature extraction techniques. Behold, this study addresses these gaps by proposing a unified experimental framework that integrates advanced preprocessing, robust feature extraction, and comprehensive evaluation. A summary of the gaps identified by this study and the respective approaches taken is given in Table 1.

Table 1. Research Gap and Proposed Approach

Research Gap	Limitations in Previous Studies	Proposed Approach in This Study
Lack of standardised preprocessing, feature extraction, and classification evaluation	Many studies apply different preprocessing, feature extraction, and classification methods in isolation, making it difficult to directly compare model performance [11–13,19].	Apply a unified pipeline (Poisson noise reduction, bilateral filtering, HOG feature extraction) to both SVM and XGBoost for a fair and controlled comparison.
Sensitivity of traditional edge detection methods	Edge detection approaches such as Sobel, Canny, and Prewitt are sensitive to noise, lighting changes, and surface texture variations [20,21].	Use noise reduction (Poisson) and edge-preserving smoothing (bilateral filter) before HOG feature extraction to improve robustness.

High computational cost and overfitting in deep learning models for limited datasets	CNN-based models require large annotated datasets, high computation, and may overfit when data is limited [2,5,6,8,22,23].	Use classical ML models (SVM, XGBoost) with HOG features for efficient computation and strong generalisation on moderate-sized datasets.

Noise reduction techniques are included in the main preprocessing route from the starting point in preparation for advanced second-phase methods that can improve the feature discriminability as well as support lower classification error rates. The first preprocessing among the techniques is the removal of Poisson Noise, which proves to be effective in scrubbing noise artifacts that are mostly due to sub-optimal illumination and inherent limitations within the sensor, and brings forth definition and clarity in structural edge information for crack identification. Next employed is a non-linear smoothing operation, Bilateral Filtering, which would intentionally reduce image noise while maintaining crucial edge features that are essential in the precise delineation of crack boundaries. This methodological approach has been featured by Zhongying He and Wen Xu [19], who have reported that using different complementary preprocessing measures together could improve crack detection accuracy while reducing false detections through the integrity of edge structure. The benefit acquired by such advanced preprocessing work is that, after that, the HOG feature extraction would actually work on optimized image data characterized.

Based on all existing methodological limitations and practical approaches, this study provides a comparative analysis of different classification frameworks for concrete bridge crack detection. The offered methodological framework integrates state-of-the-art preprocessing algorithms such as Poisson-Noise reduction and bilateral filtering with advanced feature extraction through Histogram of Oriented Gradients (HOG). This preprocessing pipeline is utilized with advanced machine learning classifiers, which are result-oriented on the SVM and Extreme Gradient Boosting (XGBoost) for effective crack detection performance. The evaluation could also include multiple dataset partitioning schemes, 5-fold cross-validation, statistical significance testing, and inference time analysis, thereby providing an all-inclusive presentation of trade-offs made by each model in terms of accuracy, robustness, and computational efficiency.

2. RESEARCH METHOD

The research methodology used is shown in Figure 1. Finding a literature review on the research is the first step in this part of the study. Once one has been found, concrete crack image data from the Mendeley data source is gathered. The pre-processing phase comes next to obtain precise data before processing. After that, the feature extraction method will be used to process the data. Labeling the characteristics and training the model are the next steps. Lastly, the model will undergo testing and evaluation.

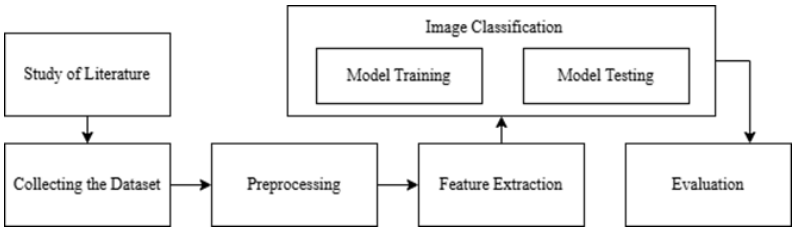


Figure 1. Research Methodology

2.1. Data Collection

For this study, an available dataset obtained from the Mendeley Data repository [2] was used; the dataset contains high-resolution images of cracked and uncracked concrete surfaces. The entire dataset has 40,000 images in JPEG/JPG format, sampled systematically into two balanced classes of 20,000 each: positive samples correspond to surfaces with cracks, while negative samples correspond to intact surfaces with no visible defects. The images are presented in RGB colour space with

standard sizes of 227 × 227 pixels; derived from 458 high-resolution source images, with all samples appropriately organised in the directories labelled with respect to their classification. Manual annotation was applied for the accurate labelling of data during the stage of dataset preparation. Figure 2 represents some samples of the dataset and the visual features and variations in both categories used in this comparative analysis.

(a) (b)

Figure 2. Samples of Concrete Dataset (a) Negative Cracked Images (b) Positive Cracked Images

To preserve the integrity of performance evaluation and maintain fidelity to the original image distribution characteristics, no additional data augmentation techniques—including rotation, horizontal or vertical flipping, scaling transformations, or contrast adjustments—were applied during the preprocessing phase. The dataset partitioning employed two evaluation schemes: the first evaluations being earned through a hold-out validation implementation involving multiple train-test distributions (50:50, 60:40, 70:30, and 80:20) and the second, following the systematic 5-fold cross-validation process. Using this example, the 70:30 setup gave 28,000 images for model training (14,000 positives and 14,000 negatives) and 12,000 images for testing (again with 6,000 positives and 6,000 negatives).

2.2. Preprocessing Data

Bilateral filtering, Poisson-noise addition, and normalization are all part of this step. Preprocessing is a step that is essential for cleaning, removing, and converting the object data type into the necessary data type. It is done before the categorization process. The objective is to improve the utilization of the data [24]. Figure 3 illustrates the steps in the data preparation pipeline.

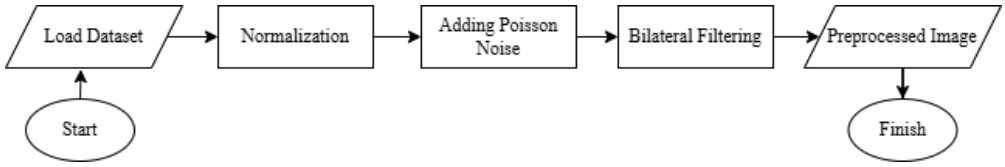


Figure 3. Image Preprocessing Stage

2.2.1. Image Normalization

Such modification in pixel lengths and their distribution alters the image normalization within its domain and has also enhanced the model's accuracy, stability, and convergence rate [25]. Further, various types of picture normalization are found to influence the performance of machine learning [26]. The major empirical procedures for photo-normalization include empirical (i.e., division by pixel intensity range value 255), min-max normalization, and z-score normalization. The empirical method attracted attention in this research.

All pixel intensities are affected by the min-max normalization method according to the formula in Equation (1):

$$x_{norm} = \frac{x_i - min(x)}{max(x) - min(x)} \quad (1)$$

In the formula above, the pixel intensity value of the image is given in addition to the minimum and maximum pixel intensities of the image and the normalized pixel intensity. Equation 2 mentioned

here simplifies the min-max normalization, specifically in the case of image intensities wherein the minimum value equals 0 and the maximum value equals 255.

$$x_{norm} = \frac{x_i}{255} \quad (2)$$

The empirical method is adopted by Equation 2. Using the mean and standard deviation from the dataset, the Z-score normalizes each pixel value in the image, scaling and shifting it. The precise process has been illustrated in Equation 3:

$$x_{norm} = \frac{x_i - \mu}{\sigma} \quad (3)$$

Where μ is the standard deviation of the image dataset, whereas σ is the mean of the image dataset (the training set).

2.2.2. Poisson Noise

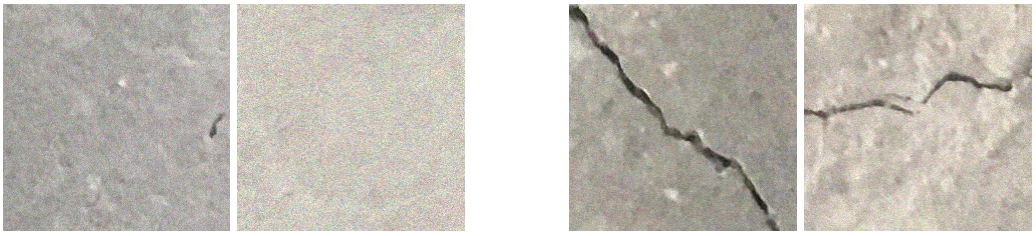
Adding noise to the training images and blurring them has become a common practice in data augmentation techniques. This sets up the model to withstand variability in the image quality used for training deep learning applications. Studies have shown that these techniques help the model learn to distinguish between signal and noise, thus improving its accuracy in performing classifications under challenging environments.

For instance, Pawara et al. applied various data augmentation techniques, such as blurring and noise addition, to develop a plant picture category. They used Convolutional Neural Networks (CNNs) servers that profited the most from this method, especially the domesticated trained-from-scratch CNNs, which can better face variability in images taken from the real world [27]. Research studies have also shown that adding noise is one of the most important factors of data augmentation, which makes it capable of solving various problems like the small training dataset or highly variable data quality and makes the models robust under diverse and degraded input images [22].

According to Zhang, the context refers to the composition of the noisy grayscale image as crack repair traces [23]. Here, Poisson noise is considered to circumvent strong variations in pixel intensities across the RGB channels during preprocessing, such as those produced with a realistic passage, such as increased model robustness. Each pixel y_i in the noisy image is assumed to follow an independent Poisson distribution, as described by:

$$P_G(y|f) = \prod_i \frac{e^{-f_i} f_i^{y_i}}{y_i!} \quad (4)$$

Where f_i represents the intensity of the original clean image; y_i is the noisy pixel value; e is the base of the natural logarithm. This probabilistic framework allows the control introduction of noise, which improves the contrast and highlights crack-related features potentially critical for detection.



(a) (b)

Figure 4. Samples of Noisy Images (a) Negative Cracked Images (b) Positive Cracked Images

2.2.3. Bilateral Filtering

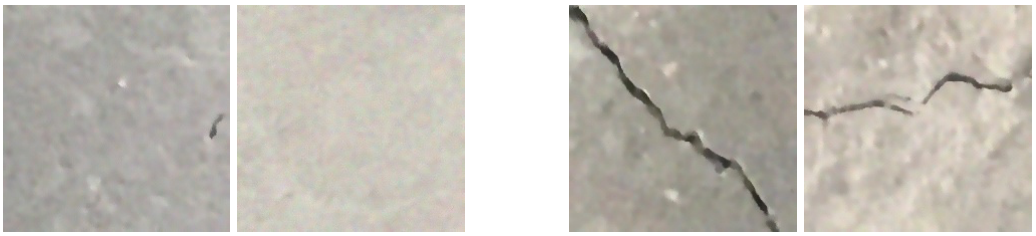
The bilateral filter is applied to the noisy images before extracting features. According to the study, bilateral filters are the best in edge preservation as compared to other image filters [28]. The bilateral filter has a general formula expressed as:

$$i_{bf}(u,v) = \frac{\sum_{x=u-\rho}^{u+\rho} \sum_{y=v-\rho}^{v+\rho} \omega_s(x,y)\omega_c(x,y)i(x,y)}{\sum_{x=u-\rho}^{u+\rho} \sum_{y=v-\rho}^{v+\rho} \omega_s(x,y)\omega_c(x,y)} \quad (5)$$

where

$$\omega_s(x,y) = exp\left\{\frac{(x-u)^2+(y-v)^2}{{\sigma_s}^2}\right\} \quad (6)$$

This means $i(x,y)$ represents the intensity of a pixel in the input image, and $i_{bf}(u,v)$ indicates the intensity of a pixel in the filtered image. The bases of ω_c and ω_s are color similarity and spatial distance, respectively. Two parameters, σ_s, and σ_c, respectively, influence their values. These were set to 75 in our experiments. In addition, ρ is set to 9. Figure 5 shows the altered image.



(a) (b)

Figure 5. Samples of Filtered Images (a) Negative Cracked Images (b) Positive Cracked Images

2.3. Feature Extraction

Practical applicability was limited for surface health monitoring techniques such as crack detection. They were slow in convergence, overfitting, costly computation, etc. [29]. Therefore, there is a need to automate a fast and feature-extracting method for processing large volumes of monitoring data.

In this study, the Histogram of Oriented Gradients (HOG) was chosen mainly for feature extraction due to its superiority as demonstrated by recent experimental studies in concrete crack detection compared to other descriptors. Several current studies have further validated HOG's excellent performance in classification, with Zhen Sun et al. reporting an accuracy of 94.38% when HOG-based feature extraction was paired with Support Vector Machine classification, which outperformed Convolutional Neural Networks, Naive Bayes, k-Nearest Neighbors, and AdaBoost algorithms under similar experimental conditions. This was also confirmed by Ida Barkiah and Yuslena Sari, who applied HOG features in Extreme Gradient Boosting with an accuracy of 96.95% and successfully solved the overfitting issue. To conclude, these empirical validations lend an impressive rationale to incorporate HOG as a strong, computationally efficient feature descriptor with consistently high output in concrete crack classification tasks. Different from other feature descriptors such as Scale-Invariant Feature Transform (SIFT), Speeded-Up Robust Features (SURF), or Local Binary Patterns (LBP), HOG has the strong compact representation highlights characteristics of the object's morphology and edges while being relatively invariant to illumination variation and shadow effects, environmental conditions routinely encountered during field applications related to structural inspection of concrete infrastructure [9, 10].

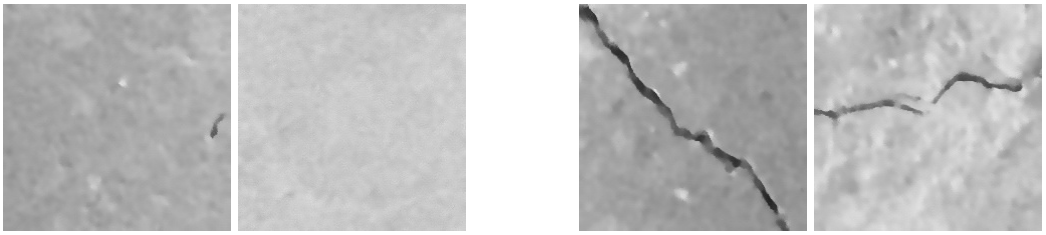
This study applies two preprocessing steps—grayscaling and HOG feature extraction—to transform raw RGB crack and non-crack images into numerical feature vectors suitable for classification.

2.3.1. Grayscaling

According to Bui et al., grayscale images improved object identification performance [30]. In the same way, Xie and Richmond discovered that grayscale images enhanced the performance of the classification method in classifying lung diseases [31]. The saturation technique is one way to convert RGB to grayscale [32], which is shown by Equation 7,

$$Y = 0.299 * R + 0.587 * G + 0.144 * B \quad (7)$$

Where Y is the pixel intensity value in grayscale photographs, illustrates how this method allocates distinct weights to components R (Red), G (Green), and B (Blue) according to their contributions to brightness or luminosity. The example, as portrayed in Figure 6, would be the result of degrees in grayscale processing.



(a) (b)

Figure 6. Samples of Grayscale Images (a) Negative Cracked Images (b) Positive Cracked Images

2.3.2. Histogram of Oriented Gradients

The HOG feature compartments the entire image into a small uniform collection and can accurately represent the distribution of intensity gradients and orientations to give a form shape locally. Moreover, normalizing several cells into a wider block region effectively reduces local illumination or shade noise [13].

Convolution filters, commonly used to extract images' edges, compute an image's vertical and horizontal gradients [13]. The following equations (8) and (9) represent the calculation of the gradient magnitude and direction, respectively:

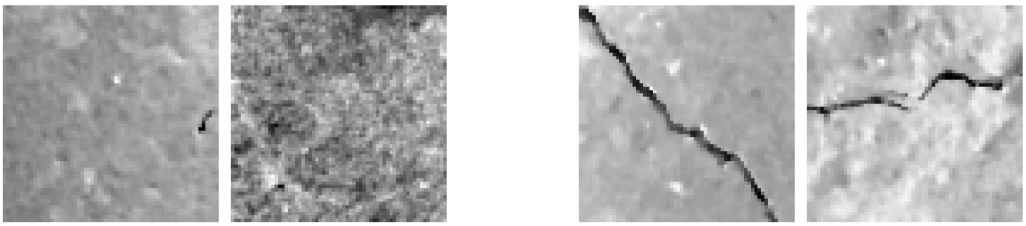
$$G = \sqrt{g_x^2 + g_y^2} \quad (8)$$

$$\theta = arctan\frac{g_y}{g_x} \quad (9)$$

The gradient in the x direction is represented by g_x, and the gradient in the y direction is represented by g_y.

The small cells that make up the image compute the HOG. It is a histogram that gives the frequency distribution for the orientation angles of the gradients and is resolved into nine bins that represent an angle from 0 to 180 degrees. Each pixel's contribution to the cell is determined by its gradient magnitude. Cells are then grouped into blocks, and all the features generated for each block are combined to yield the overall HOG features for the image [13].

This research takes the clarity-enhancing patches from original high-resolution images of 227 by 227 pixels. In this study, the images will magnify up to 64 by 64 pixels to collect feature values since HOG will require specific dimensions to compute. HOG features for a few examples of authentic images in the dataset are shown in Figure 7.



(a) (b)

Figure 7. Samples of HOG Features (a) Negative Cracked Images (b) Positive Cracked Images

2.4. Image Classification

Classification involves scientifically methodically grouping and organizing things based on their characteristics. Image categorization evolved to bridge the gap between human and computer vision by teaching the computer using data. Actual classification happens when we classify the image according to the appropriate category based on the vision content.

2.4.1. Support Vector Machine

Among the types of Artificial Neural Networks (ANN) are Support Vector Machines (SVM). The SVM classification process must comprise a training and testing phase. The objective of the SVM approach is to determine the optimum classifier function that distinguishes between two different data sets [33].

SVM uses numerous kernels to tackle problems, such as outlier detection and specifying an optimal border that defines the class of specific features. Some of these kernels incorporate the gamma parameter. The most popular kernels that use this parameter—such as sigmoid, polynomial, and radial basis function (RBF)—are presented in the equation below [14].

$$K(X,Y) = \begin{Bmatrix} (\gamma\, X \cdot Y + c)^d, \gamma > 0 & Polynominal \\ exp(-\gamma\, |X \cdot Y|^2), \gamma > 0 & RBF \\ tanh(\gamma\, X \cdot Y + c) & Sigmoid \end{Bmatrix} \quad (10)$$

Here, c represents the kernel function's variance parameter, whereas γ is the gamma parameter. X and Y correspond to the vectors from the input space. The user-definable parameters are γ, d, and c. The gamma value is computed by dividing the features in the utilized dataset by 1.

SVM is a robust classifier and regressor machine learning algorithm that seeks optimal hyperplane separation of distinct class data with maximum margin. This algorithm handles non-linearity cases satisfactorily using kernel functions like the Radial Basis Function (RBF). Thus, it works exceptionally powerfully in complicated classifications. Demonstrated in different fields, including health data, where Sabilillah et al. [24] proved its accuracy and predictive capabilities. Above all, Hasan et al. work on the classification of hyperspectral images in agricultural and urban areas, which proves that SVM with RBF kernel classifiers has an edge in superior accuracy performance in most applications [34]. The present study will utilize the SVM with RBF kernel for image classification and then compare it with the XGBoost Classifier to ascertain the performance of both classifiers.

2.4.2. XGBoost

XGBoost was introduced to the world by Friedman, which stands for gradient-boosted decision trees, with the technique now being employed to develop the GBDT algorithm (Gradient Boosting Decision Tree) [15]. While it uses techniques similar to those found in Gradient Boosting Machines (GBM), this approach is designed to optimize performance, sometimes ten times faster. Extreme Gradient Boosting (XGBoost) augments the second-order Taylor expansion of the loss function with a regularization term to find the correct answer; it minimizes decision function decrease alongside model complexity to combat better overfitting amicably [16]. XGBoost employs a

specialized approach to decision tree-based classification by utilizing a modified loss function that explicitly controls tree complexity. This loss function mathematically formulated in Equation 11 incorporates parameters such as the number of tree leaves T and leaf output scores w as input [17].

$$L_{xgb} = \sum_{i=1}^{N} L\big(y_i, F(x_i)\big) + \sum_{m=1}^{M} \Omega(h_m) \quad (11)$$

$$\Omega(h) = \gamma\, T + \frac{1}{2}\, \lambda \|w\|^2 \quad (12)$$

The trained parameters are adjusted in this research on XGBoost as follows:

1. eval_metric: It specifies the chosen evaluation metric for the impending event. In this sense, 'log loss' is used to evaluate the measurement, which reflects the ability to predict the log-likelihood that the model generates in terms of probabilities.
2. n_estimators: The number of trees to be constructed in the boosting process. The more trees, the stronger the model, but the more likely it is to overfit the data.

The most recent empirical studies on machine learning applications for classification schemes played a considerable role in the hyperparameter selection of XGBoost models. In particular, choosing 'log loss' as the evaluation criterion was justified by the work of Karin et al. [18], which proved it ideal for measuring the accuracy of the prediction probabilities in a substantial study against phishing attacks to protect bank customers. Their findings indicated that using 'logloss' actually provides a stronger mechanism by which to assess the performance of a model, as it quantifies precision on predicted class probabilities, increasing up to an impressive 94.9% accuracy. In the same way, the determination of n_estimators was guided by the study by Ida Barkiah [12], who used concrete crack monitoring, which showed that 1,200 trees were the optimal settings and got peak accuracy at 96.95%. This screening of these hyperparameters by baseline studies within this study is iteratively intended to enhance the predictive capability of the XGBoost model and reduce the risk of overfitting.

This method introduces early termination in which the regularization parameter γ specifies the minimum required loss reduction for splitting an internal node. The trees become progressively simpler with the increase of γ, so they can reduce overfitting and define better models.

Moreover, XGBoost also uses shrinkage as an additional regularization method by controlling the ratio of additive model enlargement. To do this, the contribution of each subsequent tree is reduced, which, in turn, helps to avert overfitting and increase the model's robustness. Tree complexity can be limited in several ways, including limiting tree depth and number of leaves.

2.5. Model Evaluation

A confusion matrix is an elementary statistical instrument used to evaluate algorithms in machine learning. Its typical assessment involves evaluating classification algorithms [35].

In the confusion matrix, rows and columns correspond to actual and predicted classes, respectively, while the diagonal elements indicate correctly predicted classes and misclassifications appear in off-diagonal elements [13].

Table 2. Confusion Matrix

		Prediction	
		Positive	Negative
Actual	Positive	True Positif (TP)	False Negatif (FN)
	Negative	False Positive (FP)	True Negative (TN)

Four necessary performance measures were obtained from the binary classification model's results on the testing dataset to assess the proposed crack detection technique's effectiveness fully.

In Table 2, these True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN) values are highlighted. Thereafter, these confusion matrix elements were used to determine essential performance metrics: accuracy, precision, recall, and F1-score, as per the mathematical definitions provided in Equations 13-16.

$$accuracy = \frac{T_p+T_n}{T_p+T_n+F_p+F_n} \tag{13}$$

$$precision = \frac{T_p}{T_p+F_p} \tag{14}$$

$$recall = \frac{T_p}{T_p+F_n} \tag{15}$$

$$F_1 = \frac{2\times precision\times recall}{precision+recall} \tag{16}$$

An evaluation framework would consist of different related measures. Accuracy measures how much of the sample size is accounted for by correct predictions as determined from the diagonal elements of the confusion matrix. Precision refers to the ratio of the correct classified samples to the total number of classified samples within each class. In contrast, recall or sensitivity presents the completeness of the model in terms of the calculated proportion of correctly classified sample(s) against the actual sample count in each corresponding class. It should be noted that while a high precision indicates better prediction accuracy concerning false positives, higher recall values correlate with an effective model in identifying most of the samples from their respective classes. The F1-score is the precision-recall/(precision + recall) and gives a harmonic mean of precision and recall measurement [13].

3. RESULTS AND ANALYSIS

This research utilized a computer system with the following specifications: Intel64 Family 6 Model 165 Stepping 2 GenuineIntel ~2592 Mhz, 16 GB RAM, NVIDIA GeForce GTX 1660 Ti as the VGA. The coding was done in Python 3.13.

This research uses a confusion matrix to evaluate the precision, recall, F1-score, and accuracy of feature extraction and classification. The investigative evaluation is accompanied using two different model evaluation approaches. The first approach applies the holdout method with four different training–testing split ratios: 50:50, 60:40, 70:30, and 80:20. These variations attempt to study the stability of model performance under different amounts of training data. The second approach applied 5-fold cross-validation to provide a more robust overall measure by averaging results across multiple folds.

To determine whether the performance differences between the two evaluation strategies are statistically significant, the Wilcoxon signed-rank test was performed on key performance metrics. Furthermore, an inference time analysis was performed to compare the computational efficiency of the models. This was done by measuring the average time required for a single prediction during the testing phase.

3.1. Performance Evaluation of Method

This study uses two different evaluation approaches to assess the classification performance of Support Vector Machine (SVM) and XGBoost models. These include a holdout method with four different train-to-test split ratios and 5-fold cross-validation. Evaluation metrics include accuracy, precision, recall, and F1 score.

Accuracy, precision, recall, and F1 measure model performance. This study utilizes a random state value of 42, according to the research conducted by Winanto et al [36]. The random state is a seed value used by the random number generator that determines the randomization involved in activities such as shuffling the data or random splitting of the dataset in training and testing subsets.

Random state - denotes reproducibility in the data separation process so that, throughout the experiments, the same training and test subsets can be produced by the same splitting. Table 3 illustrates the performance of both models under the holdout evaluation scheme with split ratios of 50:50, 60:40, 70:30, and 80:20.

Table 3. The Model Performance of Holdout Evaluation

Ratio	Model	Accuracy	Precision	Recall	F1-score
50:50	SVM	98.70%	98.70%	98.70%	98.70%
	XGBoost	98.66%	98.66%	98.66%	98.65%
60:40	SVM	98.72%	98.72%	98.72%	98.72%
	XGBoost	98.67%	98.68%	98.67%	98.67%
70:30	SVM	98.83%	98.83%	98.83%	98.82%
	XGBoost	98.83%	98.83%	98.83%	98.82%
80:20	SVM	98.94%	98.94%	98.94%	98.94%
	XGBoost	98.84%	98.84%	98.84%	98.84%

This table shows that both models yield high results in terms of comparison in classification performance, where the performance score from the SVM model tends to slightly exceed that of the other model in most instances. The SVM achieved its best performance with an 80:20 split, achieving an impressive score of 98.94% across all metrics, while XGBoost in the identical configuration achieved 98.84%.

The statistical test the researchers adopted to ascertain whether SVM and XGBoost really differ was the Wilcoxon signed-rank test. It showed that neither of the models examined was statistically different from the other on any of the measures. For instance, all these tests obtained p-values between about 0.10 and 0.11 for accuracy, precision, recall, and F1 score, which means that it would not be much less than 0.05; that is, a generally accepted threshold for being deemed significant in statistical terms. In other words, the models may be small along the raw numerical score differences in performance, but not to a degree that can confidently declare one truly better than the other. SVM and XGBoost were just about comparable in their performance.

Table 4. The Model Performance of 5-Fold Cross-Validation Evaluation

k-fold	Model	Accuracy	Precision	Recall	F1-score
1	SVM	98.94%	98.94%	98.94%	98.94%
	XGBoost	99.01%	99.01%	99.01%	99.01%
2	SVM	98.78%	98.78%	98.78%	98.77%
	XGBoost	99.05%	99.05%	99.05%	99.05%
3	SVM	98.76%	98.76%	98.76%	98.76%
	XGBoost	98.88%	98.88%	98.88%	98.87%
4	SVM	98.42%	98.42%	98.42%	98.43%
	XGBoost	98.66%	98.66%	98.66%	98.66%
5	SVM	98.88%	98.88%	98.88%	98.88%
	XGBoost	98.93%	98.93%	98.93%	98.93%

The findings from 5-fold cross-validation were delineated in Table 4. In most of the folds, XGBoost performed marginally better than SVM, but in the second fold, XGBoost recorded the highest accuracy of 99.05%. SVM showed its mediocre performance and scored consistent values across folds for 98.42% to 98.94%.

The Wilcoxon signed-rank test was similarly applied to evaluate the cross-validation performance differences between SVM and XGBoost models. The statistical analysis yielded identical p-values of 0.0625 across all performance metrics—accuracy, precision, recall, and F1-score—which did not reach the conventional significance threshold of 0.05. However, these variables were marginally close to the cut-off threshold compared to the holdout evaluation. It indicates a trend towards slight superiority of XGBoost over SVM in cross-validation, although the difference was not considered strong enough to be significant.

Essentially, both evaluation approaches confirm that SVM and XGBoost perform at a similarly high level for this classification task. However, subtle performance variations between the two methods will be discussed further in the next section, where a more direct comparison between the evaluation methods will be conducted, along with an analysis of inference times.

3.2. Comparison between SVM and XGBoost Method

This section compares both models based on the following three perspectives: (i) evaluation methods' effect on performance metrics, (ii) statistical significance testing of these differences, and (iii) inference time trade-offs.

First, Table 5 presents an overview of the average performance of SVM and XGBoost applied to holdout and 5-fold evaluations. Both models achieved very high and stable classification results with insignificant differences between methods.

Table 5. Evaluation Comparison of SVM and XGBoost

Model	Evaluation	Accuracy	Precision	Recall	F1-score
SVM	Holdout (avg)	98.80%	98.80%	98.80%	98.80%
XGBoost	5-Fold CV	98.75% ± 0.0018	98.76% ± 0.0018	98.75% ± 0.0018	98.75% ± 0.0018
SVM	Holdout (avg)	98.75%	98.75%	98.75%	98.75%
XGBoost	5-Fold CV	98.83% ± 0.0015	98.83% ± 0.0015	98.83% ± 0.0015	98.83% ± 0.0015

The Wilcoxon signed rank test was performed independently for each model and each metric to statistically analyze whether the type of evaluation method (holdout vs. cross-validation) had any effect on the performance of the models. The test resulted in finding that none of the differences were statistically significant ($p=1.0$ for all metrics), indicating that both methods give essentially equivalent performances of SVM and XGBoost on this dataset.

Table 6. Inference Time per Image

Model	Inference Time per Image (sec)
SVM	0.009066
XGBoost	0.001285

From a functional consideration point of view, inference time is pertinent most of the time for almost all resource-constrained environments and real-time purposes. As presented in Table 6, XGBoost accomplishes significantly faster inference per image (0.001285 seconds), compared to SVM (0.009066 seconds). Going by this difference, the times are approximately seven times faster. In applications where time is critical, this is indeed significant even for applications that have similar classification accuracies.

Importantly, the confusion matrices in Figure 8 from the qualitative analyses at the 80:20 split show more about the decision tendencies of the classifiers. SVM has been said to have a lower false positive rate in comparison to false negative, reflecting somewhat a conservative form of classification, favoring not to have, but rather be satisfied by missing some cracked surfaces. In effect, XGBoost leaned over to more FPs rather than FN, denoting an aggressive approach seeking cracks. Such behavior might be beneficial for safety-critical applications in which undetected damage may be more expensive than a few false alarms.

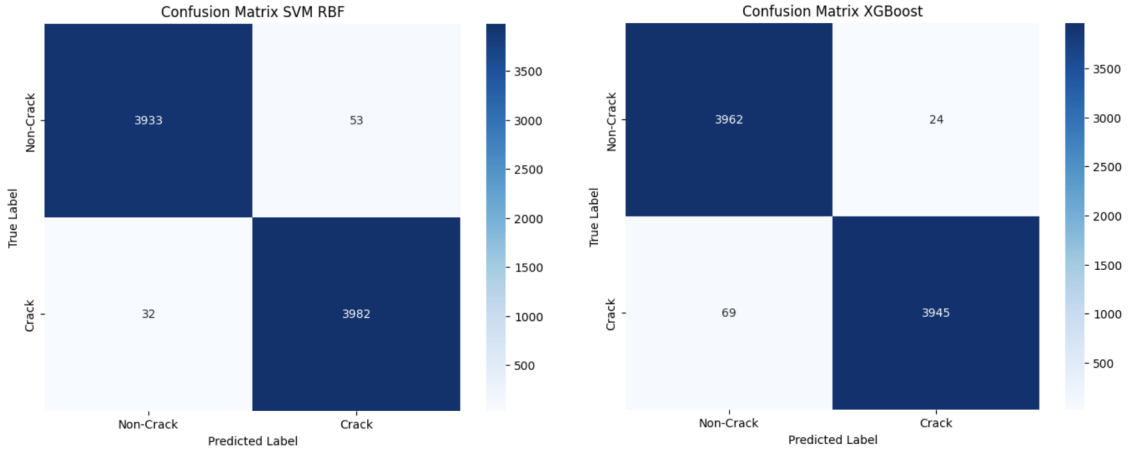


Figure 8. Confusion Matrix of 80:20 Data Composition (a) SVM (b) XGBoost

Necessarily, although statistical analyses demonstrate equal accuracy for all pairs of evaluation methods and models, practical reasonings tell a great deal about trade-offs. XGBoost offers a substantial inference speed advantage and heightened sensitivity, whereas SVM provides a slightly more conservative classification pattern. The choice between these models may therefore depend on the operational priorities of the deployment context.

4. CONCLUSION

This study investigates the performance comparison between Radial Basis Function Support Vector Machine and XGBoost classifiers in the comfort of an automated detection of cracks in concrete images using Histogram of Oriented Gradients (HOG) features extraction, followed in combination with two preprocessing approaches named Poisson Noisy Image Generation and Bilateral Filtering. Both models achieved extremely high classification accuracy across the holdout and 5-fold cross-validation evaluation protocols, with the best performance being as high as 98.94% for SVM and 99.05% for XGBoost. Statistical analysis with the Wilcoxon signed-rank indicated there are no significant differences in accuracy between evaluation methods or between the two classifiers, showing that they are equally good on the given dataset.

The analysis of confusion matrices demonstrated that SVM and XGBoost showed different classification behavior: SVM showed a rather conservative error profile, being inclined towards fewer false positives, whereas XGBoost was aggressive in terms of high sensitivity to cracked surfaces and their detection. Such performance discrepancies signal that model selection must hinge on application priorities—when false alarms present heavy costs, SVM may be chosen, whereas XGBoost might be favored for safety-critical applications where maximum crack detection is sought. In addition, from the inference time measurements, XGBoost was found to be almost seven times faster than SVM, which can tip the scales favorably for real-time implementation.

Among the limitations of this study is a specific dependence on a certain preprocessing pipeline: Poisson Noise and Bilateral Filtering. Yet, there is one method of extracting the specific feature, as HOG. This combination proved extremely useful in this study; there is a possibility that another preprocessing or feature-extraction technique, be it deep feature representations from convolutional neural networks, may further enhance the results. In addition, the present dataset only consisted of static images under controlled conditions, with not much evaluation of the performance under other environmental factors: lighting changes, surface contamination, image blur, or real-time acquisition scenarios.

Future research directions should also include the study of alternative preprocessing methods such as adaptive histogram equalization, Gaussian blur, and edge-preserving filtering, combined with hybrid extraction techniques that can combine conventional descriptors such as Local Binary Patterns or Gabor Filters with HOG feature descriptors to see if they can help improve classification

performance. However, widening the comparative study to a larger spectrum of classifiers with Random Forest, k-Nearest Neighbor, and deeper architectures like the ResNet, DenseNet, EfficientNet, or MobileNetV3 could provide deeper insights into the performance of the various algorithms. Then it would be very important to calibrate all the previous models in terms of real-time video processing or on-site inspection systems to assess their robustness and reliability under practical constraints and dynamic conditions.

REFERENCES

[1] A. Kadir, A. S. Sukri, N. H. Aswad, Masdiana, and Nasrul, “Evolution and Implications of Changes in Seismic Load Codes for Earthquake Resistant Structures Design,” Civil Engineering Journal, vol. 10, no. 1, pp. 62–82, Jan. 2024, doi: https://doi.org/10.28991/cej-2024-010-01-04.

[2] L. Zhang, F. Yang, Y. Daniel Zhang, and Y. J. Zhu, “Road crack detection using deep convolutional neural network,” IEEE Xplore, Sep. 01, 2016. https://ieeexplore.ieee.org/abstract/document/7533052.

[3] I. N. Sutarja, I. G. A. Susila, and M. A. A. Sumekar, "Strengthening strategy of old bridge in Indonesia (Tukad Yeh Bakung) by using optimization and preservation approach," J. Civ. Constr. Environ. Eng., vol. 6, no. 2, pp. 46–53, Mar. 2021, doi: 10.11648/j.jccee.20210602.13.

[4] A. N. A. Thohari, A. Karima, K. Santoso, and R. Rahmawati, "Crack detection in building through deep learning feature extraction and machine learning approach," J. Appl. Inform. Comput., vol. 8, no. 1, pp. 1–6, Jul. 2024, doi: 10.30871/jaic.v8i1.7431.

[5] H. K. Shin, Y. H. Ahn, S. H. Lee, and H. Y. Kim, “Automatic Concrete Damage Recognition Using Multi-Level Attention Convolutional Neural Network,” Materials, vol. 13, no. 23, p. 5549, Dec. 2020, doi: https://doi.org/10.3390/ma13235549.

[6] S. Li and X. Zhao, “Automatic Crack Detection and Measurement of Concrete Structure Using Convolutional Encoder-Decoder Network,” IEEE Access, vol. 8, pp. 134602–134618, 2020, doi: https://doi.org/10.1109/access.2020.3011106.

[7] L. Wu, "Applications of computer vision technologies of automated crack detection and quantification for the inspection of civil infrastructure systems," Ph.D. dissertation, Dept. Civil Engineering, Univ. of Central Florida, Orlando, FL, USA, 2015. [Online]. Available: https://stars.library.ucf.edu/etd/1320.

[8] Q. Yuan, Y. Shi, and M. Li, “A Review of Computer Vision-Based Crack Detection Methods in Civil Infrastructure: Progress and Challenges,” Remote Sensing, vol. 16, no. 16, pp. 2910–2910, Aug. 2024, doi: https://doi.org/10.3390/rs16162910.

[9] L. Meng, Z. Wang, Y. Fujikawa, and S. Oyanagi, "Detecting cracks on a concrete surface using histogram of oriented gradients," in Proc. Int. Conf. Adv. Mechatronic Syst. (ICAMechS), Beijing, China, 2015, pp. 103–107, doi: 10.1109/ICAMechS.2015.7287137.

[10] M. R. K., A. S., and Y. Mohana, "Inspection, identification and repair monitoring of cracked concrete structure – an application of image processing," in Proc. 3rd Int. Conf. Commun. Electron. Syst. (ICCES), Coimbatore, India, 2018, pp. 1151–1154, doi: 10.1109/CESYS.2018.8723898.

[11] Z. Sun, E. Caetano, S. Pereira, and C. Moutinho, “Employing histogram of oriented gradient to enhance concrete crack detection performance with classification algorithm and Bayesian optimization,” Engineering Failure Analysis, vol. 150, p. 107351, Aug. 2023, doi: https://doi.org/10.1016/j.engfailanal.2023.107351.

[12] I. Barkiah and Y. Sari, “Overcoming Overfitting Challenges with HOG Feature Extraction and XGBoost-Based Classification for Concrete Crack Monitoring,” International Journal of Electronics and Telecommunications, pp. 571–577, Jun. 2023, doi: https://doi.org/10.24425/ijet.2023.146509.

[13] H. Zoubir, M. Rguig, M. El Aroussi, A. Chehri, and R. Saadane, “Concrete Bridge Crack Image Classification Using Histograms of Oriented Gradients, Uniform Local Binary Patterns, and Kernel Principal Component Analysis,” Electronics, vol. 11, no. 20, p. 3357, Oct. 2022, doi: https://doi.org/10.3390/electronics11203357.

[14] V. Vapnik, The Nature of Statistical Learning Theory. New York: Springer, 1995.

[15] H. Hasan, H. Z. Shafri, and M. H. Habshi, "A comparison between support vector machine (SVM) and convolutional neural network (CNN) models for hyperspectral image classification," in IOP Conf. Series: Earth Environ. Sci., vol. 357, 2019, doi: 10.1088/1755-1315/357/1/012080.

[16] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," The Annals of Statistics, vol. 29, no. 5, pp. 1189–1232, 2001.

[17] Y. Chen, X. Wang, Y. Jung, V. Abedi, R. Zand, M. Bikak, and M. Adibuzzaman, "Classification of

short single-lead electrocardiograms (ECGs) for atrial fibrillation detection using piecewise linear spline and XGBoost," Physiol. Meas., vol. 39, no. 10, p. 104006, 2018, doi: 10.1088/1361-6579/aadf0f.
[18] C. Bentéjac, A. Csörgő, and G. Martínez-Muñoz, "A comparative analysis of XGBoost," arXiv, 2019, doi: 10.48550/arXiv.1911.01914.
[19] Z. He and W. Xu, "Deep learning and image preprocessing-based crack repair trace and secondary crack classification detection method for concrete bridges," Structure and Infrastructure Engineering, pp. 1–17, Mar. 2024, doi: https://doi.org/10.1080/15732479.2024.2330702.
[20] I. Abdel-Qader, O. Abudayyeh, and M. E. Kelly, "Analysis of Edge-Detection Techniques for Crack Identification in Bridges," Journal of Computing in Civil Engineering, vol. 17, no. 4, pp. 255–263, Oct. 2003, doi: https://doi.org/10.1061/(asce)0887-3801(2003)17:4(255).
[21] S. Dorafshan, R. J. Thomas, and M. Maguire, "Benchmarking image processing algorithms for unmanned aerial system-assisted crack detection in concrete structures," Infrastructures, vol. 4, no. 2, p. 19, May 2019, doi: 10.3390/infrastructures4020019.
[22] D. V. L. F. Fernandes, L. D. A. Tavares, and F. R. C. Silva, "A review of deep learning techniques for medical image analysis," J. Imaging, vol. 9, no. 2, pp. 46, 2023, doi: 10.3390/jimaging9020046.
[23] K. Alomar, H. I. Aysel, and X. Cai, "Data augmentation in classification and segmentation: A survey and new strategies," J. Imaging, vol. 9, no. 6, pp. 195, 2023, doi: 10.3390/jimaging9060195.
[24] F. T. Sabilillah, C. A. Sari, R. B. Abiyyi, and A. Susanto, "Comparison of machine learning algorithms on stunting detection for 'Centing' mobile application to prevent stunting," Sinkron: Jurnal dan Penelitian Teknik Informatika, vol. 8, no. 4, pp. 2360–2368, Oct. 2024, doi: 10.33395/sinkron.v8i4.13967.
[25] X. Zhong, B. Gallagher, K. Eves, E. Robertson, T. N. Mundhenk, and T. Y.-J. Han, "A study of real-world micrograph data quality and machine learning model robustness," npj Computational Materials, vol. 7, no. 1, p. 212, Oct. 2021, doi: 10.1038/s41524-021-00616-3.
[26] M. M. Ali, S. A. S. Asad, and S. A. R. Naqvi, "Design and development of a smart agriculture system using IoT," Proc. IEEE 2nd Int. Conf. Recent Trends in Electrical & Information Technologies (RTEICT), Noida, India, 2016, pp. 145–150, doi: 10.1109/RTEICT.2016.7808140.
[27] P. Pawara, E. Okafor, L. Schomaker, and M. Wiering, "Data augmentation for plant classification," in Advances in Intelligent Systems and Computing, vol. 744, A. B. M. S. Ali, Ed. Cham, Switzerland: Springer, 2017, pp. 499–506, doi: 10.1007/978-3-319-70353-4_52.
[28] F. Zhang, "Research on image denoising," M.S. thesis, Hangzhou Dianzi University, Hangzhou, China, 2017, doi: 10.7666/d.D01269744.
[29] R. Fan and N. Dahnoun, "Real-time stereo vision-based lane detection system," Measurement Science and Technology, vol. 29, no. 7, p. 074005, 2018, doi: 10.1088/1361-6501/aac94e.
[30] X. Yao, "Evolutionary artificial neural networks," Int. J. Neural Syst., vol. 4, no. 3, pp. 203–222, 1993, doi: 10.1142/S0129065793000282.
[31] H. M. Bui, M. Lech, E. Cheng, K. Neville, and I. S. Burnett, "Using grayscale images for object recognition with convolutional-recursive neural network," in Proc. IEEE, Piscataway, NJ, USA, 2016.
[32] Y. Xie and D. Richmond, "Pre-training on grayscale ImageNet improves medical image classification," in Proc. European Conf. Computer Vision (ECCV) Workshops, Munich, Germany, 8–14 Sep. 2018.
[33] Y. P. Pasrun, M. Muchtar, A. N. Basyarah, and Noorhasanah, "Indonesian license plate detection using morphological operation," in IOP Conf. Series: Mater. Sci. Eng., Institute of Physics Publishing, Jun. 2020, doi: 10.1088/1757-899X/797/1/012037.
[34] D. H. Abd, A. T. Sadiq, and A. R. Abbas, "Classifying political Arabic articles using support vector machine with different feature extraction," in Proc. Int. Conf. Appl. Comput. Support. Industry: Innovation Technol., Springer, pp. 79–94, 2019.
[35] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," Inf. Process. Manag., vol. 45, no. 4, pp. 427–437, 2009, doi: 10.1016/j.ipm.2009.03.002.
[36] E. A. Winanto, Y. Novianto, S. Sharipuddin, I. S. Wijaya, and P. A. Jusia, "Peningkatan performa deteksi serangan menggunakan metode PCA dan Random Forest," J. Teknol. Inf. dan Ilmu Komputer, vol. 24, no. 11, 2024, doi: 10.25126/jtiik.2024117678.