Baghdad Science Journal

Volume 23 | Issue 3 Article 28

3-25-2026

Leveraging Feature Fusion and Convolutional Neural Networks for Concrete Crack Prediction

Amal Abdulbaqi Maryoosh
Department of Computer Engineering, Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran, a.abdolbaghi@tabrizu.ac.ir

Saeid Pashazadeh
Department of Information Technology, Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran, pashazadeh@tabrizu.ac.ir

Pedram Salehpour
Department of Computer Engineering, Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran, psalehpoor@tabrizu.ac.ir

How to Cite this Article

Maryoosh, Amal Abdulbaqi; Pashazadeh, Saeid; and Salehpour, Pedram (2026) "Leveraging Feature Fusion and Convolutional Neural Networks for Concrete Crack Prediction," *Baghdad Science Journal*: Vol. 23: Iss. 3, Article 28.
DOI: https://doi.org/10.21123/2411-7986.5253

BAGHDAD SCIENCE JOURNAL 2026;23(3):1068–1083

Scan the QR to view the full-text article on the journal website

RESEARCH ARTICLE

Leveraging Feature Fusion and Convolutional Neural Networks for Concrete Crack Prediction

Amal Abdulbaqi Maryoosh [1], **Saeid Pashazadeh** [2,*], **Pedram Salehpour** [1]

[1] Department of Computer Engineering, Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran
[2] Department of Information Technology, Faculty of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran

ABSTRACT

Detection of cracks in concrete is crucial for the safety of bridges and the overall infrastructure. This paper presents a new hybrid method that combines handcrafted and deep features to significantly improve classification accuracy. Texture and semantic information are captured using Local Binary Patterns (LBP) and a pre-trained Xception model, respectively. These features are converted by the Bag of Visual Words (BoVW) method, combined, and the best features are selected by using the Apriori algorithm. The selected features are classified utilizing the light MobileNetV3, a large network. Our method is tested on four public datasets, namely CODEBRIM, DIMEC, Crack, BCD, and Bridge, using 10, fold cross, validation. The model we introduced has drastically diminished the error rate by less than 1%. Furthermore, it performed well on the accuracy metric, achieving scores of 0.9995, 0.9983, 0.9998, and 0.9993, respectively. Precision values reached 0.9998, 1.0000, 1.0000, and 0.9991, while recall figures stood at 0.9995, 0.9938, 0.9998, and 0.9986 for these same datasets. Compared to other deep learning models trained on the same datasets, our model shows highly encouraging and promising outcomes.

Keywords: Association rule mining, BoVW, CNN, Feature fusion, Feature selection

Introduction

Bridges are essential structures for transportation. They are exposed to deterioration and damage due to rust, chemical degradation, unfavorable loading, erosion factors, or natural disasters such as earthquakes, floods, and hurricanes throughout their service lives, so their efficiency must be maintained. Due to the huge costs of building bridges, periodic maintenance must be done on bridges to ensure the health of their structures, identify damage, and carry out appropriate maintenance activities. Several structural health monitoring techniques have been proposed to assess the health of bridges and avoid major accidents caused by bridge collapses. The assessment of the health condition of the bridge and all its elements is carried out by defining a series of data obtained from visual inspections, which allows for the drawing up of damage maps to work on. However, there are instances where visual inspection may be difficult or impossible, particularly in critical areas of bridges like roofing, corners, and highs. [1,2]

One of the main reasons why researchers have been investigating the use of advanced technologies for damage detection is the pressing need to have accurate and automated methods of identifying damage. As computer vision, machine learning (ML), and deep learning (DL) have been evolving at a very fast pace, vision-based approaches have become promising tools for the automation of bridge damage classification. Even though they can be very effective, these systems are still exposed to challenges, which are usually huge. Structural photos normally have issues such as low contrast, crack patterns that are not uniform, and complicated backgrounds, which all together make it very difficult to detect accurately. [3,4]

Received 6 April 2025; revised 18 July 2025; accepted 29 July 2025.
Available online 25 March 2026

* Corresponding author.
E-mail addresses: a.abdolbaghi@tabrizu.ac.ir (A. A. Maryoosh), pashazadeh@tabrizu.ac.ir (S. Pashazadeh), psalehpoor@tabrizu.ac.ir (P. Salehpour).

https://doi.org/10.21123/2411-7986.5253

Traditional image processing and ML methods highly depend on handcrafted features like texture, shape, and color, but they usually miss capturing high-level semantic information and don't perform well under changing lighting and surface conditions. However, deep convolutional neural networks (CNNs) provide end-to-end learning and are able to extract better features from raw images, but on the other hand, they require huge amounts of labeled data, are prone to overfitting, require high computational resources, and are low in interpretability. [5,6]

Besides that, simple feature methods are usually limited in their accuracy and robustness. By combining several types of features, feature fusion has shown better results, but at the same time, it brings redundancy and raises the computational cost. Therefore, good feature selection methods are required to get rid of noise and unhelpful features, and to improve classification efficiency. [7,8]

This paper tries to tackle the problems by suggesting a hybrid classification framework that merges feature fusion, feature selection, association rule mining, and deep learning. We highlight our works' contributions in the following points:

- Use reliable hybrid supervised and unsupervised learning techniques that drastically enhance error rate performance over existing methods/baseline models. Construct a new algorithm based on feature fusion, Apriori association rule mining, and CNN for reliable concrete damage classification.
- We demonstrate the proposed method's performance on COncrete Defect BRidge IMage (CODEBRIM), the DIMEC-crack database, Bridge Crack Detection (BCD), and bridge datasets.

Related works

Feature fusion

Feature fusion of local and deep features holds the key to significantly improved classification performance. By integrating local features that store fine details, along with deep features derived from neural networks, a more thorough data depiction is obtained. The fusion of these modalities results in the development of more accurate and consistent models that understand and examine complex image data proficiently. Several research studies in the recent past have merged different feature extraction approaches to increase the performance of image classification in various domains. For example, Shrinivasa S., et al. [9] combined handcrafted and deep features to facilitate the improvement of classification accuracy in scene image classification. The standard ResNet model retrieves the deep features of the images, while the Binary Robust Invariant Scalable Key Points (BRISK) descriptor retrieves the handcrafted local features. Fusing the two different kinds of image features creates a new type of image feature known as the hybrid feature. Thereafter, they utilize a Support Vector Machine (SVM) training model to classify scene images. BRISK is a robust invariant key point descriptor. While effective for certain tasks, its performance might vary across different types of scene images. While SVMs are powerful, they can be computationally expensive for enormous datasets. Their performance can also be sensitive to the choice of kernel and hyperparameters. Ahmed, et al. [10] combined SIFT and CNN features in remote sensing image classification. They integrated these features SVM as the basis for classification. Similar to BRISK, SIFT is a strong local descriptor but may have limitations in certain remote sensing contexts (e.g., highly textured areas with ambiguous keypoints). Other local features might offer complementary information.

The objective of this fusion strategy is to use the distinct benefits of each feature extraction method while reducing their particular constraints. In order to classify breast biopsy images, Tripathi, et al. [11] constructed a patch-based classifier by combining a CNN with BoVW. ResNet50, DenseNet169, and InceptionV3 evaluated their technique using the BACH-2018 challenge dataset. In their work, they show a method that outperforms all three models collectively. Their approach may be quite good at identifying local features; however, a patch-based method could overlook some of the global contextual information present in the whole biopsy image. It is possible that the integration of the patch, level BoVW with CNN features, still offers room for improvement through the inclusion of additional global image features. For pavement crack detection, Chen, et al. [12] employed a fusion of LBP and principal component analysis (PCA). They used LBP to extract features from each frame of the road video, PCA to reduce the features' dimension, and SVM to train the samples. LBP is excellent for texture, but pavement cracks can also have distinct shapes, edges, and contextual information that LBP alone might not fully capture. While PCA is common, it's a linear dimensionality reduction technique. It might not optimally preserve discriminative information for complex non-linear crack patterns compared to non-linear methods.

Ozkay S., et al. [13] obtained a feature vector using deep and textural feature extraction techniques. In the first line, they extracted the feature from AlexNet and generated the feature vector using the FC8 layer; in the second line, they applied LBP to generate the feature vector and applied it again to the resulting

feature vector, then combined the two lines to obtain the feature vector; after that, they utilized the neighborhood component analysis (NAC) algorithm for feature selection. Finally, train the selected features using SVM. For the METU crack dataset, the developed model has achieved 99.9% accuracy in classifying crack images. Applying LBP twice to the resulting feature vector after the first LBP application is unusual, and its precise benefit or potential drawbacks (e.g., increased redundancy, loss of original information) are not immediately clear from the description. Although this very high accuracy is impressive, it can sometimes indicate potential overfitting or a relatively "easy" dataset unless it is carefully validated or tested on diverse datasets.

Concrete damage prediction

This section presents a full description of the most recent models that identify damage in concrete. We have noted different methods in the studies published on concrete damage detection. Some studies are based on deep learning models only, whereas others emphasize machine learning techniques.

Also, there is a portion of research that combines image processing techniques with deep learning or machine learning to achieve better detection results. The literature on concrete damage detection shows that there are many different types of approaches that researchers have used in their work. Some studies focus on deep learning models, while others leverage machine learning techniques. Interestingly, a notable number of investigations also combine image processing techniques with either deep learning or machine learning to enhance their detection capabilities. For instance, a combination of multi-layered image preprocessing (containing homomorphic filtering, median filtering, Gaussian filtering, and the Otsu method) with CNN proposed by Fu et al.[14] for crack image identification and crack position detection was applied to the 2042 crack image and the 8680 background image. To find surface cracks, Zhang F. et al.[15] used three machine learning algorithms: SVM, random forest, and k-nearest neighbor. They also used image processing techniques: min-max gray level discrimination and Laplace sharpening, and dimensionality reduction methods: F-Score Feature Selection and Principal Component Analysis. They discovered that the V-SVM machine learning method can achieve an accuracy of 0.8899, PCA can reduce the number of features to 0.95 of their original variance, and Laplace sharpening can enhance the image's color. Bhalaji K., et al.[16] utilized 12 transfer learning DCNN models to classify cracks in three publicly accessible datasets: CCIC, SDNET, and BCD. Due to the limited efficiency of transfer learning DCNN models on the SDNET images, two image enhancement approaches, LBP and contrast enhancement, were utilized on the images. In addition, the SVM is trained using deep features obtained from the last fully connected layer of the DCNNs. Detection accuracy was improved in all DCNN-dataset combinations by integrating deep features with SVM. To enhance the effectiveness of damage detection techniques in the Structural Monitoring and Control (SMC) benchmark bridge, Ahmadian et al.[17] employed XGBoost to assess the significance of features derived from the sensor data. They trained the stacking approach, a combination of four machine-learning algorithms (SVM, k nearest neighbor, decision tree, and Gaussian Naive Bayes), using the time frame selected features. The stacking approach obtained the maximum performance, with an average accuracy rate of 0.931. The study of Golding et al.[18] proposed a deep learning-based crack detection method using the CNN technique. Before training the pre-trained VGG16 architecture to construct several CNN models, the authors processed the images from the CCIC dataset to enhance the performance of CNN classification. Image processing employs thresholding, grayscale, and edge detection methods for crack detection, but deep learning does not use them. The analysis revealed that grayscale models are on par with RGB models; however, training significantly accelerates the performance enhancement of the former. The thresholding and edge detection models lagged behind RGB models in terms of performance. The researchers infer that deep learning approaches for crack detection are color-independent; deep learning crack detection does not depend on color. Zadeh et al.[19] incorporated four deep learning models to crack detection of the CCIC dataset. Specifically, they leveraged the power of pre-trained deep learning architectures, such as ResNet50, VGG19, EfficientNetV2, and Inception V3, to fine-tune their approaches. The selection of these models was based on their demonstrated performance and flexibility in the field of image analysis. Among the models evaluated, EfficientNetV2 demonstrated superior performance in terms of increasing efficiency and reducing false positives. In the area of transfer learning, Islam et al.[20] used four well-known deep learning models: DenseNet161, VGG16, AlexNet, and ResNet18. They used weights that had already been trained on the ImageNet dataset. The models were tested utilizing the CCIC and BWCI datasets. The recommendations were using AlexNet to implement augmentation techniques, which involved deleting the last three fully connected (FC) layers and substituting them with an FC2 layer to align the output features for binary classification. The researchers

concluded that the suggested AlexNet model outperformed the other models they used. Yang et al.[3] proposed utilizing the DCNN method to classify cracks in three public datasets: CCIC, BCD, and SDNET. The suggested approach transfers three types of knowledge from previous research accomplishments: sample knowledge, parameter knowledge, and model knowledge. The VGG network evolved additional fully linked layers as a new learning framework. The researchers confirmed the validity and effectiveness of the suggested method. Bukhsh, et al.[21] evaluated a combination of in-domain and cross-domain transfer learning strategies for bridge damage detection. Six publicly accessible visual inspection datasets (SDNET, CDs, ICCD, BCD, CODEBRIM, and MCDS) were used in extensive comparisons to examine the effects of cross-domain and in-domain transfer, with different starting procedures, for binary and multiclass classification. The pre-trained models were also assessed for their capacity to handle the incredibly limited amount of data available. The results demonstrate that using cross-domain and in-domain transfer consistently yields better performance, particularly when dealing with small datasets. Zoubir et al.[22] investigated the concrete damage detection method using more than 3,000 images of concrete surfaces, both cracked and uncracked, showing various crack patterns and concrete surface conditions. To characterize the images, they used a combination of Histogram of Oriented Gradients (HOG) and Uniform Local Binary Patterns (ULBP) features. In order to lessen the number of features, they decided to use Kernel Principal Component Analysis (KPCA). For the classification task, they experimented with three different machine learning models. Their results demonstrated that the subsequent combination of features from the two methods, HOG and ULBP (once subjected to KPCA), with an SVM classifier brought about the most superior result, getting an extremely high accuracy of 99. 26%. Zoubir, with the other team,[23] has released a large-scale, publicly accessible dataset that includes over 6,900 annotated images of cracked and non-cracked concrete bridges and culverts. This dataset is extremely valuable as it presents difficult surface conditions and various crack sizes and patterns. They performed an evaluation of the dataset by three state-of-the-art Deep Convolutional Neural Networks (DCNNs), namely VGG16, VGG19, and InceptionV3, utilizing a transfer learning method. These models were able to distinguish between cracked and non-cracked images, with the peak testing accuracy being 95. 89%. The findings indicate that the dataset has a significant capacity for training deep learning networks that are specialized in the detection of concrete cracks in bridge structures. Our model, compared to existing ones, offers a broader and well-ordered hybrid method by merging handcrafted features (LBP) and deep semantic features (from Xception), combined with mid-level encoding via BoVW and then Apriori-based feature selection. Prior art only dabbled in different mixes of fusion and classifiers (e.g., SIFT + CNN + SVM or BRISK + ResNet + SVM), our work brings in association rule mining (Apriori) as a feature selection tool, which is something not seen before and quite effective in this case. Different from earlier works, which are mainly plagued with feature redundancy, poor generalization, or heavy computation, our method is more transparent. By keeping only, the frequent and significant feature combinations, it lowers the risk of overfitting. Thus, the generalization capability standard is notably raised for a number of benchmark datasets. Our experiments validate that our approach has gained better accuracy and precision (99.8%) on four varied datasets, which would be hard to match by many deep and hybrid models that have been published so far. These findings testify to the originality and applicability of our paper on the automated detection of cracks in concrete structures, especially in the case of real bridge monitoring systems, where dependability and resource saving are key factors.

Datasets

We tested our model on four concrete damage image datasets. Three of them are related to bridge damage or crack detection, and the fourth one is for building crack detection.

BCD Dataset: Xu, et al.[24] released a dataset with 2,068 images equally representing bridge cracks and non-cracks. The images were taken by a Phantom 4 Pro's CMOS surface array camera at a resolution of 1024 × 1024 pixels. These images, after a two-stage reduction process first to 512 × 512 and then to 224 × 224 dimensions, were used to form the final dataset. In the end, this dataset comprises 4,057 images of cracks and 2,013 images of non-cracks.

Bridge Datasets: Zoubir, et al.[22] compiled a dataset containing 1,304 cracked and 1,806 non-cracked RGB bridge images of 200 × 200 pixels each. These photos show concrete surfaces/window cracks, etc., and have been taken during real bridge inspections. Before converting the images to grayscale to get rid of crack, like noise, they were subjected to a 3 × 3 median filter as a pre-processing step.

CODEBRIM Dataset: CODEBRIM includes images of 30 distinct bridges, excluding bridges without defects introduced by Mundt, et al.[25] as a novel, concrete defect bridge image dataset. The bridges were selected based on their levels of general

degradation, severity, number of defects, and surface appearance, such as roughness and color. Images were taken in various weather conditions, including wet or dirty surfaces, with multiple cameras at different magnifications. Because the defects were very small, high-resolution images were necessary. A drone (unmanned aerial vehicle, UAV) captured part of the dataset as some sections of a bridge were inaccessible. A set of four different high-resolution cameras with large focal lengths took the dataset while changing the distance, angle, and perspective of the pictures. Besides that, to uniformly illuminate the darkest sections of the bridge, they used a diffused flash. The supplementary material provides detailed information about the specific camera models and their respective technical specifications. The collection contains 7,455 high-resolution photos of 30 different bridges at various sizes and resolutions. We divided it into two classes: defects and background. The defects have 4,388 images, while the background contains 3,067 images.

DIMEC-Crack Database: Lopez D., et al.[26] created a new dataset specifically for crack semantic segmentation. This dataset comprises 10,092 high-resolution images (1920 × 1080 pixels) extracted from video footage of various concrete bridges, captured by a UAV equipped with high-resolution cameras. While their original study utilized 96 × 96 pixel patches, we plan to use the full, raw images. The dataset is divided into two classes: 7,872 images featuring cracks and 2,220 images without cracks.

Materials and methods

The first step is applying the preprocessing steps to the datasets, then two data types will be extracted using the Xception model and LBP. We extracted the mid-level feature representation for each type of feature using the BoVW method in five sizes of visual words (50, 100, 200, and 300). We use the BoVW method because it enhances generalizability. Then, we concatenate the two types of features. After fusion, we may encounter feature duplication, which reduces classification accuracy and increases data bias. So, we apply the Apriori algorithm to reduce the number of features and eliminate duplication. Finally, we train the data using MobileNetV3-Large. We will explain the proposed model in this section. Fig. 1, illustrates the general structure of the proposed model.

Preprocessing

The preprocessing steps may increase model inference and reduce model training time. Reducing the size of the input images can significantly reduce the model's training time while preserving its performance, especially if there are many images. This paper employs the following preprocessing techniques:

- Image resizing is the process of increasing or decreasing the size of an image while maintaining its original properties. It works to make the model faster. The large image is difficult to process effectively. Therefore, we resize the different sizes of the dataset's images to 100 × 100.
- Grayscale image: To simplify the image's data and reduce algorithmic processing requirements, we convert the images to grayscale.
- Removing duplicate data is considered a fundamental operation in data cleaning and preparation. It eliminates biases in data and enhances classification accuracy.
- Data augmentation techniques: To increase the size of a dataset, enhance the model's generalization, and reduce overfitting, we use data augmentation techniques to produce new images from existing ones. We used the albumentations library's augmentation methods (HorizontalFlip, ShiftScaleRotate, VerticalFlip, Rotate).
- The data normalization technique sets pixel intensity values to a predefined range, usually between 0 and 1. It enhances the model's effectiveness and reduces its computation.

Feature extraction

This paper uses two features: texture features (low-level features) extracted by Local Binary Pattern (LBP) and deep features extracted by the Xception model. This section briefly explains each type of them.

Local binary pattern

The LBP is a textural descriptor described by Ojala, et al.[27] and utilized for classification. To calculate a binary pattern, LBP compares the intensity of a center pixel (threshold) with the intensity of the eight neighbor pixels around it. LBP may be formalized as follows in Eqs. (1) and (2) for a center pixel I_c and an adjacent pixel I_i (i = 1, 2, …, P)[28]:

$$LBP = \sum_{i=1}^{P} F(I_i - I_c).\ 2^{i-1} \tag{1}$$

$$F\,(I) = \begin{cases} 1, & I \geq 0, \\ 0, & otherwise. \end{cases} \tag{2}$$

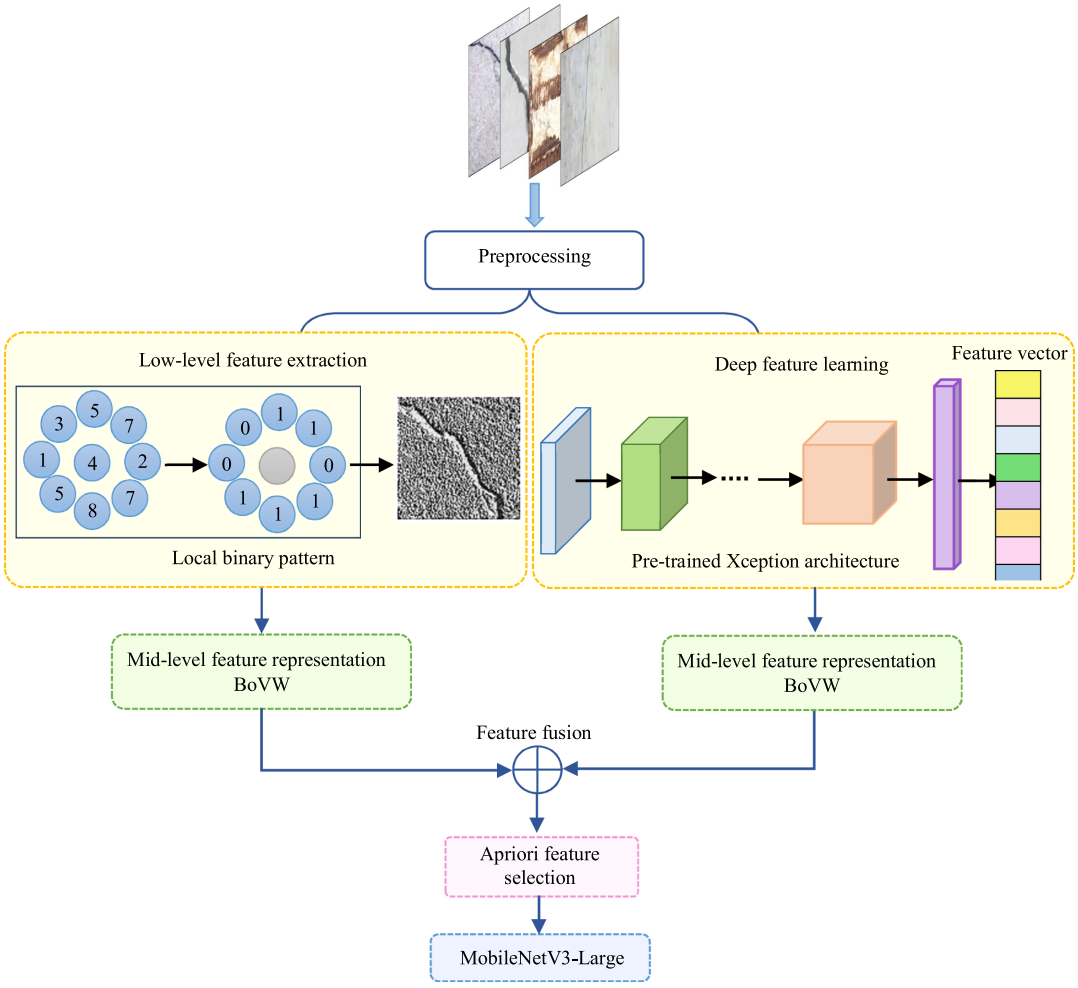


Fig. 1. The general structure of the proposed method.

Where P represents the number of neighbors considered, F(I) is a thresholding function, and 2^{i-1} represents the weight factor used to give each neighbor a unique binary place value. LBP is highly effective in capturing local micro-patterns, such as cracks, textures, and edges, and it is proven robust to illumination changes. Since photos from bridge inspection are typically characterized by varying lighting and diverse surface features, LBP provides a fast and reliable technique to encode the fine texture details required for the detection of cracks. The straightforwardness of LBP fits perfectly with the complexity of the deep features.

Xception model

The Xception model[29] that was used in this research was initially trained on the ImageNet dataset, a large-scale dataset composed of natural images. The Xception, which is recognized for its high efficiency with depthwise separable convolutions, has been able to deliver the highest level of performance among various image classification tasks, including the distinguishing of fine details. The Xception model utilizes its final MaxPooling2D layer to obtain features with the size of (3 × 3 × 1024). After that, we put a flattened layer to squeeze the feature array dimension into a one-dimensional array. The feature vector obtained from the flattened layer is of the (1 × 9216) type. Retrieving features from an intermediate layer of Xception enables us to get a rich, generalized representation of the image content that the model from a massive dataset (ImageNet) has learned. It offers very strong contextual information that goes beyond the local textures. We picked Xception because it offers a good balance of accuracy and computational efficiency, making it appropriate for our feature extraction requirements.

BoVW

A BoVW is an overarching mechanism to derive a single representation of the entire image from a set of local features. Various ways based on BoVW are

commonly used in computer vision because they are highly precise, fast, and easy to implement. In the first place, it was a tool for natural language processing (NLP) and text retrieval, later changing to embrace computer vision. The BoVW method has recently gained notoriety for its excellent performance in semantic image analysis and image classification tasks across numerous benchmark datasets. The BoVW features are obtained by first training the feature vector patches with a k-means clustering algorithm. Next, we decide on the dictionary size (k), which is the number of clusters, and take the cluster centroids as the visual words.[30] The steps of BoVW are given as follows:[31]

1. Decide how many patches should be used to split each feature vector into patches.
2. In order to find out the dictionary size (k), we tried 50, 100, 200, and 300, which correspond to the number of K, meaning clusters.
3. Locate the center of each cluster. These centers are the visual words. The dimension of the visual word vector is the same as K.
4. We created local feature vectors by calculating histograms of each feature vector.

BoVW is a method to create mid-level features, LBP describes local pixel patterns, and Xception offers high, level semantic context. In contrast, BoVW is a connecting step that takes the raw local features (e.g., the LBP responses of different patches or regions) and maps them into a semantic space using a visual vocabulary. Hence, we can combine local visual hints to form a completely global image descriptor that statistically summarizes the visual patterns. At the same time, it is very helpful to reduce the loss of spatial detail, which is one of the drawbacks of strictly global features, and at the same time, it provides a much richer descriptor for the complicated damage patterns.

Feature fusion

After texture and deep feature extraction, feature selection, and mid-level feature representation, we will gather these features together. The total feature sizes will be (100, 200, 400, and 600). Feature fusion is intended to capture more unique features, give semantic information for describing the relevant parts of images, and, at the same time, remove the irrelevant ones. Through these effective feature extraction methods and a thorough feature selection process, our model can fully capture tiny details as well as the overall semantics of bridge inspection images, thereby making it a more robust and accurate damage classification system.

Apriori algorithm for feature selection

After fusing the diverse feature sets, we became aware of the possibility of high dimensionality, which could even include redundant or less informative features. As a result, we decided to use an Apriori, based feature selection method. This algorithm assists us in discovering patterns and relationships that frequently co, occur in the combined features. Through the use of Apriori, we are able to single out and keep the most discriminative and correlated feature combinations, thus decreasing redundancy, facilitating computational efficiency, and most importantly, raising the overall interpretability and predictive power of our final classifier. This stage is paramount for resolving the typical problem of feature fusion leading to dimensionality explosion and making sure that only genuinely relevant information is used. Apriori detects frequent itemsets (sets of features) that happen together in a dataset. When considering feature selection, frequent itemsets can be viewed as combinations of features that have a strong association with the target class. The idea is to keep features that regularly coexist with certain labels. Thinking of feature selection, we can liken the features of the dataset to “items” and each record of data to a “transaction.” By identifying frequent itemsets (i.e., frequently occurring combinations of features), we can spot features that commonly co-occur, which may signify significant relationships or redundancies. We rely on two fundamental approaches in feature selection:

1. Redundancy Reduction: If we find that large frequent itemsets (frequent features) are always present together, it might suggest that frequent features are highly correlated or redundant. Thus, we choose to keep only one of them.
2. Filtering by Support/Confidence: We use thresholds for support and confidence to filter out less significant features or rules. Features involved in rules that don’t meet these thresholds might be considered less important.

Feature extraction is mainly carried out to achieve two goals. Firstly, by choosing just a few highly informative features from the original data, the classifier can perform more efficiently. Secondly, the features that do not truly represent the data object are eliminated so that the accuracy of the classification can be increased. In the experiment, we set min- support = 0.9 and min- confidence = 0.9 as thresholds for choosing features. It is clear from the experimental results that the feature selection with the Apriori algorithm is responsible for the 17.39% boost in the classification accuracy. The model, which had an accuracy of 82.56% and a loss

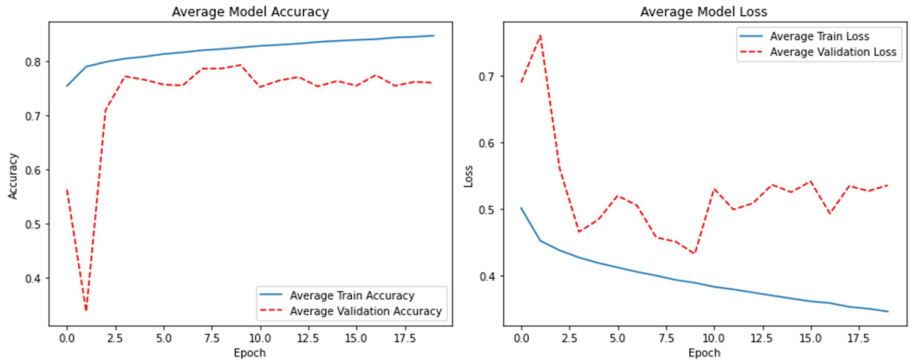


Fig. 2. The accuracy and loss plot before using Apriori feature selection for feature size 100.

Table 1. Number of features before and after Apriori feature selection for the CODEBRIM dataset.

	No. of the features
Before Apriori feature selection	100
After Apriori feature selection	47, 31, 17, 42, 9, 27, 34, 28, 41, 26, 100, 17, 25, 12, 15, 29, 72, 42, 88, 21

of 17.44% before the application of the Apriori algorithm, is now more accurate and less faulty. Fig. 2, presents these performance metrics by means of accuracy and loss plots derived from 10, fold cross, validation.

After applying the Apriori algorithm to the features derived from BoVW, the number of features selected varies across different instances. This is because Apriori selects only those features that meet the minimum support and confidence thresholds, which are not uniformly satisfied by all features in each instance. Consequently, the number of retained features is not consistent across the dataset. Table 1 illustrates this.

Table 1 explains the number of features for 20 instances after the Apriori feature selection of the CODEBRIM dataset. It also reveals that before feature selection each example originally had 100 BoVW features. The number of features selected by Apriori for each instance varies drastically, from only 9 features to as many as 100 features, depending on how many features satisfied the selection criteria in each case. This variation necessitates the use of imputation to standardize input dimensions for the proper functioning of models. Since model training requires a fixed feature size per example, data was treated for missing values arising from this inconsistency using the mean imputation method. This approach substitutes missing values with the mean value of the respective feature from the whole dataset, thus maintaining the data structure and limiting information loss or bias.

MobileNetV3_Large

Google specifically developed MobileNet as an open-source computer vision model for classifier training. It is a convolutional neural network that uses depthwise convolutions to greatly decrease the parameter count compared to other networks, thereby creating a lightweight deep neural network. MobileNet is a compact model designed to accommodate limited resources, such as low latency and low power consumption. They are specifically parameterized to cater to various use cases, making them well-suited for mobile applications and other situations with constrained resources. Depthwise separable convolutions enable MobileNet to achieve efficiency by reducing the number of multiply-accumulates and the model's overall size. Although MobileNet exhibits superior speed and compactness compared to other prominent networks, it may sacrifice some accuracy compared to bigger, resource-intensive models. Nevertheless, its performance remains commendable, with just a little decline in accuracy.[32] In our work, we trained the utilized datasets using MobileNetV3-Large. It is more accurate and has a lower latency than MobileNetV2.

Experimental results

Experimental environment

In this section, we demonstrate that the proposed model is generalizable and capable of producing reliable results by evaluating their performance on four datasets using 10-fold cross-validation.

We experimented with our code, which was implemented in the following environments: the TensorFlow and Keras frameworks. We developed it in Python 3.9 using the Anaconda Navigator IDE and utilized it with Windows 10 64-bit. An i7-8550U CPU runs at 1.99 GHz and 32 GB of RAM, and the experiment's settings are as follows: the batch size is 128,

20 epochs, and the optimization method for loss is Adam optimization.

Evolution metrics

In order to assess our method, we make use of various performance metrics, such as accuracy, precision, recall (or sensitivity), F1, score, AUC score, and error rate. All of these metrics are derived from the confusion matrix, which depicts the classifier's performance. This matrix consists of the four basic elements for binary classification:

- True Positive (TP): It is the count of actual positive cases that have been correctly identified as positive.
- True Negative (TN): It is the count of actual negative cases that have been correctly identified as negative.
- False Positive (FP): It is the count of actual negative cases that have been wrongly identified as positive.
- False Negative (FN): It is the count of actual positive cases that have been wrongly identified as negative. These elements serve as the basis for calculating the following evaluation metrics, and the formulas for each metric are explained as follows, from Eqs. (3) to (9):[33]

1. Accuracy (Overall Correctness): Accuracy refers to how often a model classifies samples correctly in relation to the total number of samples. This is a very intuitive concept; however, it tends to be misleading in the case of imbalanced datasets.

$$\text{Accuracy} = \frac{(\text{TP} + \text{TN})}{(\text{TP} + \text{TN} + \text{FN} + \text{FP})} \tag{3}$$

2. Precision (Positive Predictive Value): Precision measures the ratio of positive cases that have been correctly identified out of all the cases that have been predicted as positive. High precision is important in a scenario where the cost of a false positive is high, for instance, not performing unnecessary maintenance on a healthy bridge.

$$\text{Precision} = \frac{\text{TP}}{(\text{TP} + \text{FP})} \tag{4}$$

3. Recall (Sensitivity or True Positive Rate, TPR): Recall is the measure of actual positive cases that have been correctly identified. It is especially important to have high recall in the case of bridge damage detection, as overlooking actual damage (false negatives) may lead to very dangerous situations.

$$\text{Recall} = \frac{\text{TP}}{(\text{TP} + \text{FN})} \tag{5}$$

4. F1-Score (Harmonic Mean of Precision and Recall): The F1-Score is a single value indicator that considers both precision and recall. It is particularly helpful in situations where there is a significant disparity between classes or when it is equally important to reduce both false positives and false negatives.

$$\text{F1} - \text{score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

5. Error Rate (Overall Incorrectness): Error rate is inversely related to accuracy, and it reflects the percentage of total samples that have been classified inaccurately.

$$\text{Error Rate} = 1 - \text{Accuracy} = \frac{(\text{FP} + \text{FN})}{(\text{TP} + \text{TN} + \text{FP} + \text{FN})} \tag{7}$$

6. Specificity (True Negative Rate - TNR): Specificity determines the fraction of real negative cases that are correctly recognized. This is crucial for making sure that the non-damaged cases are correctly identified.

$$SP = \frac{TN}{TN + FP} \tag{8}$$

7. AUC: The AUC score reflects how well the model can separate positive and negative classes at different classification thresholds. A model with a higher AUC can better distinguish between classes overall.

$$AUC = \frac{SP + SE}{2} \tag{9}$$

Results and discussion

This section outlines a detailed breakdown of the classification results that the proposed model achieved on the datasets used. We optimized the model training for four features, sized variations on each of the datasets, which differed depending on the size of the BoVW. Table 2 shows the outcomes of the training of the proposed model using four different datasets.

We have conducted the experiments, and our findings were compared with the details given in Table 2. Our results show that the size of the visual feature,

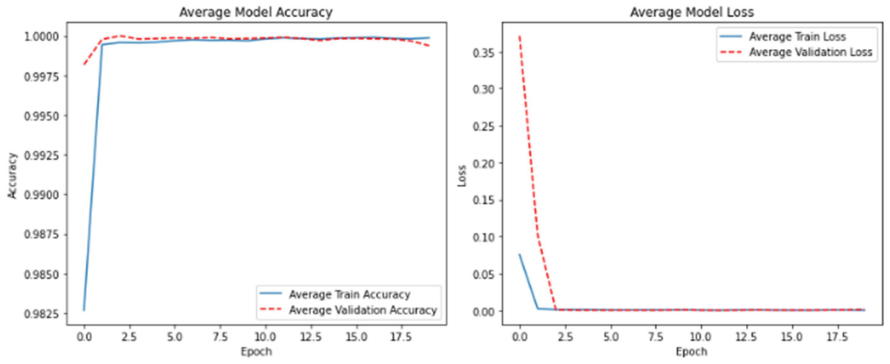
Table 2. Performance of the proposed method with varying numbers of visual words for CODEBRIM and DIMEC-Crack datasets.

		Number of visual words			
Datasets	Performance Metrics	100	200	400	600
CODEBRIM	Accuracy	**0.9995**	0.9993	0.9989	0.9984
	Precision	**0.9998**	0.9995	0.9987	0.9985
	Recall	**0.9995**	0.9992	0.9994	0.9989
	F1-score	**0.9996**	0.9994	0.9991	0.9987
	ROC-AUC	**0.9996**	0.9993	0.9988	0.9984
	Error rate	**0.0005**	0.0007	0.0011	0.0016
DIMEC-Crack	Accuracy	**0.9983**	0.9963	0.9902	0.9953
	Precision	**1.0000**	1.0000	1.0000	0.9982
	Recall	**0.9938**	0.9864	0.9640	0.9847
	F1-score	**0.9969**	0.9929	0.9814	0.9913
	ROC-AUC	**0.9969**	0.9932	0.9820	0.9920
	Error rate	**0.0017**	0.0037	0.0098	0.0047
BCD	Accuracy	**0.9998**	0.9997	**0.9998**	0.9990
	Precision	**1.0000**	0.9999	**1.0000**	0.9995
	Recall	**0.9998**	0.9996	**0.9997**	0.9990
	F1-score	**0.9999**	0.9998	**0.9999**	0.9992
	ROC-AUC	**0.9999**	0.9997	**0.9999**	0.9990
	Error rate	**0.0002**	0.0003	**0.0002**	0.0010
Bridge	Accuracy	**0.9993**	0.9980	0.9963	0.9940
	Precision	**0.9991**	0.9976	0.9990	0.9921
	Recall	**0.9986**	0.9952	0.9874	0.9862
	F1-score	**0.9988**	0.9964	0.9932	0.9891
	ROC-AUC	**0.9991**	0.9972	0.9935	0.9916
	Error rate	**0.0007**	0.0020	0.0037	0.0060

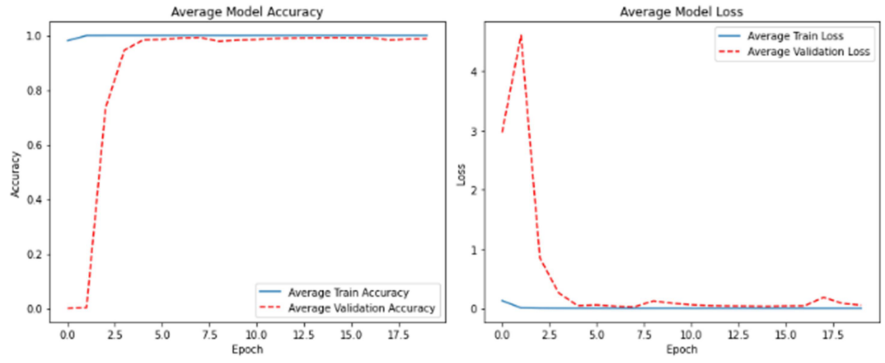
which is of 100, has a dominant performance over all the datasets that were tested. The size of the features of 100 is well balanced, as it has enough descriptive power and at the same time does not introduce noise or redundancy. The best results were obtained for all data sets by using 100 features. The model at this size manifested an ideal capturing of the real underlying patterns that result in a minimization of both false positives and false negatives, as indicated by the high precision, recall, and F1 metrics. If the feature number goes beyond 100, we mostly see a flattening of the efficiency or a slight but noticeable lowering in some of the metrics, notably the accuracy and F1 score. This is consistent with the notion that the more features you add beyond the optimal amount, the more redundant or less discriminative pieces of information you are likely to bring in. Our Apriori, based feature selection method helps to reduce this effect. However, an excessively large pool of initial features can still result in increased model complexity, possible overfitting, and longer training/inference times without corresponding gains in performance.

With the CODEBRIM dataset, using 100 features alone led to an accuracy peak of 0.9995, along with a very high precision of 0.9998 (meaning false positives were greatly minimized), a recall of 0.9995, an F1 score of 0.9996 (that harmonized precision and recall), an ROC, AUC of 0.9996, and an exceptionally low error rate of 0.0005. The DIMEC, Crack dataset performance was also the highest with 100 features, producing an accuracy of 0.9983, a flawless precision of 1.0000, a recall of 0.9938, an F1 score of 0.9969, an ROC AUC of 0.9969, and an error rate of 0.0017. A precision of one here means the model is perfect in terms of not confusing non-cracked areas with cracked ones. Likewise, the BCD dataset with 100 features showed remarkable results, including 0.9998 accuracy, 1.0000 precision (zero false positives), 0.9998 recall (only 0.0002 false negatives), 0.9999 for both F1 score and ROC, AUC, and an extremely low error rate of 0.0002. The optimized feature set was a major factor in the performance of this dataset. Last but not least, the Bridge dataset also performed the best with 100 features, scoring an accuracy of 0.9993, a precision of 0.9991, a recall of 0.9986, an F1 score of 0.9988, an ROC, AUC of 0.9991, and an error rate of 0.0007. This profound inspection of how feature size influences performance truly reveals that 100The consistently high performance across all evaluated metrics highlights the effectiveness of our proposed feature fusion and selection strategy.

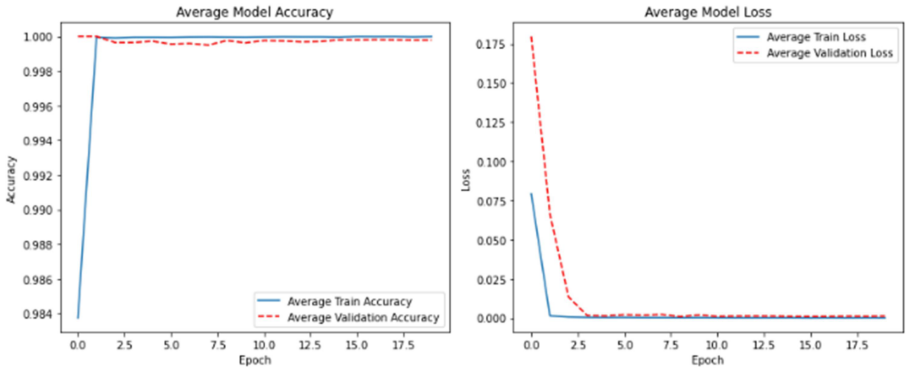
Fig. 3 shows the plot of the average of the best accuracy and loss for all datasets. Whereas Fig. 4 shows the average of the confusion matrix of the best results for all datasets.



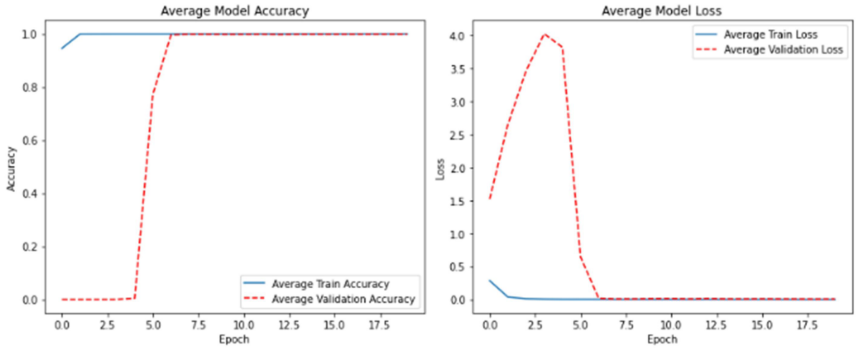
(a)



(b)



(c)



(d)

Fig. 3. Visual representation of the average accuracy and loss obtained from 10-fold cross-validation for the optimal results on each dataset: (a) CODEBRIM, (b) DIMEC-Crack, (c) BCD, and (d) Bridge, all at a feature size of 100.

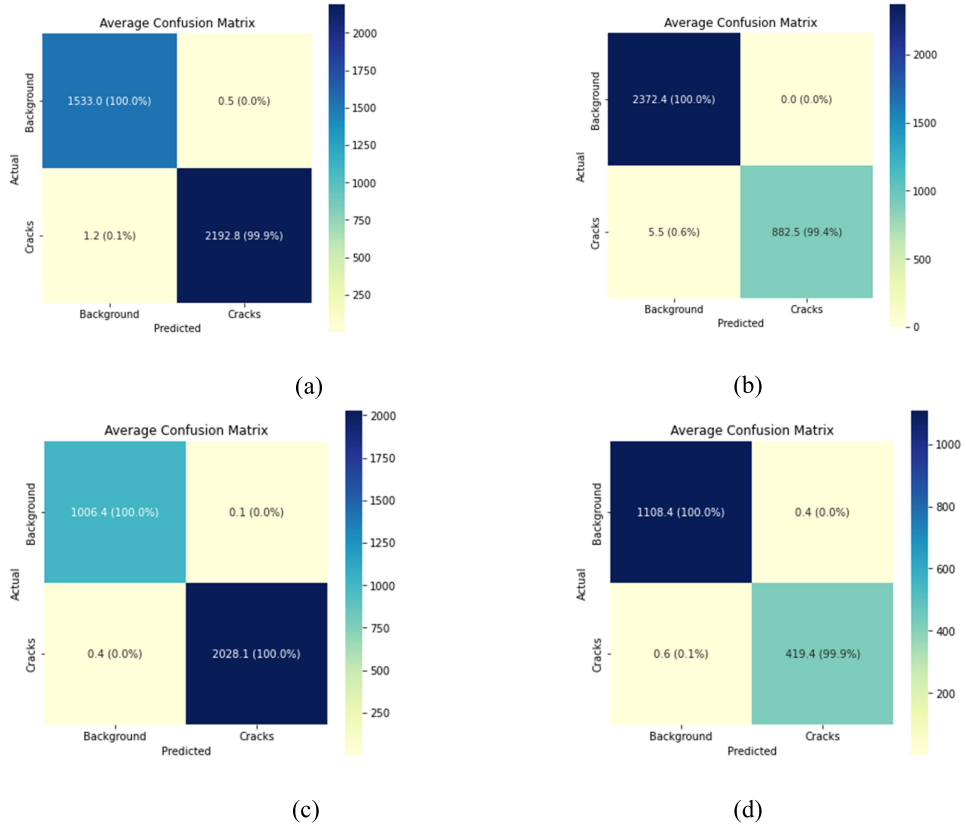


Fig. 4. Average confusion matrices reflecting the optimal performance across 10-fold cross-validation for: (a) CODEBRIM, (b) DIMEC-Crack, (c) BCD, and (d) Bridge datasets, all with a feature size of 100.

We compared the performance of the proposed model with some of the studies that classify the same datasets. Table 3 presents the accuracy scores for each dataset and compares them with related studies.

As shown in Table 3, The study conducted by Bhalaji K., et al.[16] tested the BCD dataset using transfer learning DCNN and SVM. They employed 12 transfer learning methods: The VGG16 model achieved an accuracy of 0.9983, with nearly more than 0.99 for each precision, recall, and F1-score; the VGG19 model achieved an accuracy of 0.9967, with nearly more than 0.99 for each precision, recall, and F1-score; the Xception model achieved an accuracy of 0.9967, with precision, recall, and F1-score reaching $>$ 0.99; the ResNet 101 model achieved an accuracy of 0.995, with a precision of 1.0000, a recall of 0.99, and an F1-score of 0.995; the ResNet 152 model achieved an accuracy of 0.9983, with approximately $>$ 0.99 for each precision, recall, and F1-score. InceptionV3 achieved an accuracy of 0.9983, a precision of 0.99, a recall of 0.99, and an F1-score of 0.99. InceptionResNet V2 achieved an accuracy of 0.995 with a precision of 1.0000, a recall of 0.99, and an F1 score reaching 0.995. MobileNet achieved an accuracy of 0.9983, surpassing 0.99 in each of the precision, recall, and F1-score categories. MobileNetV2 achieved an accuracy of 0.9983 with a precision, recall, and F1 score of $>$ 0.99. DenseNet 121 achieved an accuracy of 0.9967 with a precision, recall, and F1-score of 1.0000, while the EfficientNetB0 model yielded an accuracy of 0.996 and a precision of 0.9983, with a precision, recall, and F1-score of $>$ 0.99.

Yang, et al.[3] also used transfer learning, utilizing 13 layers of VGG16 and 2 FC layers. They trained their proposal with the BCD dataset, and they achieved an accuracy of 0.9972, a precision of 0.9646, and an AUC of 0.9999. Bukhsh, et al.[21] employed transfer learning using in-domain, cross-domain, and combination strategies. They tested the BCD dataset for in-domain, achieving an accuracy of 0.975 with a precision of 0.979, a recall of 0.975, and an F1-score of 0.976; for cross-domain, they achieved an accuracy of 0.986 with a precision of 0.987, a recall of 0.986, and an F1-score of 0.986. The combination strategy yielded the highest performance, achieving an accuracy of 0.989, a precision of 0.99, a recall of 0.989, and an F1-score of 0.989. Xu, et al.[24] proposed a model based on the ASPP module and depthwise separable convolution. They trained the BCD dataset on 300 epochs, achieving an accuracy of 0.9637 with

Table 3. Performance metrics comparison across four datasets for different methods.

Ref.	Method	k-fold CV	No. of epochs	Performance Metrics	Datasets			
					BCD	Bridge	CODEBRIM	DIMEC-Crack
16	Transfer learning DCNN and SVM	-	-	Accuracy	0.9983	-	-	-
3	Transfer learning VGG	-	20	Accuracy	0.9972	-	-	-
				Precision	0.9646			
				ROC-AUC	0.9999			
21	Cross-domain TL	-	30	Accuracy	0.976	-	-	-
				Precision	0.981			
				Recall	0.976			
				F1-score	0.977			
	In-domain TL			Accuracy	0.986			
				Precision	0.987			
				Recall	0.986			
				F1-score	0.986			
	Combination			Accuracy	0.989			
				Precision	0.990			
				Recall	0.989			
				F1-score	0.989			
24	atrous convolution, ASPP, and depthwise separable convolution	-	300	Accuracy	0.9637	-	-	-
				Precision	0.7811			
				Sensitivity	1.0000			
				Specificity	0.9583			
				F1-score	0.8771			
22	HOG + ULBP + KPCA+ SVM	5	-	Accuracy	-	0.9926	-	-
				Precision		0.99		
				Recall		0.9923		
				F1-score		0.9912		
23	VGG16	5	10	Accuracy	-	0.9489	-	-
	VGG19					0.9539		
	InceptionV3					0.9589		
MobileNetV3_ Large		10	20	Accuracy	0.8214	0.7259	0.7753	0.9151
				Precision	0.9112	0.3000	0.9271	0.9970
				Recall	0.8305	0.0029	0.6825	0.6158
				F1-score	0.8620	0.0057	0.7677	0.7613
				ROC-AUC	0.8168	0.5014	0.7955	0.8076
Proposed Method		10	20	Accuracy	0.9998	0.9993	0.9995	0.9983
				Precision	1.0000	0.9991	0.9998	1.0000
				Recall	0.9998	0.9986	0.9995	0.9938
				F1-score	0.9999	0.9988	0.9996	0.9969
				ROC-AUC	0.9999	0.9991	0.9996	0.9969

a precision of 0.7811, 1.0000 of recall, and an F1-score of 0.8771.

In their study, Zoubir, et al.[22] used ULBP, HOG, KPCA, and SVM to classify concrete cracks in the self-made bridge dataset. They achieved an accuracy of 0.9926, a precision of 0.95, a recall of 0.9923, and an F1-score of 0.9912. Zoubir with another team[23] utilized the Bridge dataset, supplemented it with culvert images, and applied transfer learning to the CNN models. They achieved accuracies of 0.9489 with VGG16, 0.9539 with VGG19, and 0.9589 with InceptionV3.The InceptionV3 model produced the most favorable outcomes. When we compared our proposal to previous studies, we found that it outperformed in all feature sizes.

Conclusion

This paper proposes an automated concrete crack classification framework that effectively uses a mix of local handcrafted texture features, deep semantic representations, mid-level visual encoding, and association rule-based feature selection. The proposed model achieves state, of, the, art performances on several benchmark datasets by integrating Local Binary Patterns (LBP), Bag of Visual Words (BoVW), Xception deep features, and the Apriori algorithm, trained with the MobileNetV3 Large architecture. The model showed a very high average classification accuracy of more than 99.8% and a very low average error of less than 1%, which are signs of its robustness

and efficiency. This work validates the significance of combining different types of features and the importance of feature redundancy reduction for higher model accuracy and generalization. The proposed model system enhances the predictive accuracy and keeps the computational complexity at a low level, which opens the possibility of it being operated in environments with limited resources. Future works will be geared towards assessing the model's ability to generalize to real-world, unseen data and its potential expansion in the wider field of structural health monitoring. This study can also be extended to develop real-time crack detection systems, where the on-site analysis might be done using edge computing. Moreover, a further examination of different feature fusion techniques and advanced feature selection methods might lead to even better results.

Authors' declaration

- Conflicts of Interest: None.
- We hereby confirm that all the Figures and Tables in the manuscript are ours. Furthermore, any Figures and images that are not ours have been included with the necessary permission for republication, which is attached to the manuscript.
- No animal studies are present in the manuscript.
- No human studies are present in the manuscript.
- Ethical Clearance: The project was approved by the local ethical committee at University of Tabriz, Iran.

Authors' contribution statement

A.A.M.: Writing – original draft preparation, validation; methodology, investigation, formal analysis and conceptualization. S.P.: Writing – review and editing, project administration, formal analysis, conceptualization and supervision. P.S.: Writing – review and editing, supervision and investigation.

References

1. Belcore E, Di Pietra V, Grasso N, Piras M, Tondolo F, Savino P, *et al.* Towards a FOSS Automatic Classification of Defects for Bridges Structural Health Monitoring. ASITA. 2022:298–312. https://doi.org/10.1007/978-3-030-94426-1_22.
2. Qiao W, Ma B, Liu Q, Wu X, Li G. Computer Vision-Based Bridge Damage Detection Using Deep Convolutional Networks with Expectation Maximum Attention Module. Sensors (Basel). 2021;21(3). https://doi.org/10.3390/s21030824.
3. Yang Q, Shi W, Chen J, Lin W. Deep convolution neural network-based transfer learning method for civil infrastructure crack detection. Autom Constr. 2020;116. https://doi.org/10.1016/j.autcon.2020.103199.
4. Ali L, Alnajjar F, Jassmi HA, Gocho M, Khan W, Serhani MA. Performance Evaluation of Deep CNN-Based Crack Detection and Localization Techniques for Concrete Structures. Sensors (Basel). 2021;21(5). https://doi.org/10.3390/s21051688.
5. Eltahir MM, Aldehim G, Almalki NS, Alnfiai MM, Osman AE. Reinforced concrete bridge damage detection using arithmetic optimization algorithm with deep feature fusion. AIMS Mathematics. 2023;8(12):29290–306. https://doi.org/10.3934/math.20231499.
6. Liu D, Liu Y, Li S, Li W, Wang L. Fusion of handcrafted and deep features for medical image classification. J Phys: Conf Ser. 2019;1345(2):022052. https://doi.org/10.1088/1742-6596/1345/2/022052.
7. Zhu Q, Zhong Y, Liu Y, Zhang L, Li D. A deep-local-global feature fusion framework for high spatial resolution imagery scene classification. Remote Sensing. 2018;10(4):568. https://doi.org/10.3390/rs10040568.
8. Azam F, Rizvi A, Khan WZ, Aalsalem MY, Yu H, Zikria YB. Aircraft classification based on PCA and feature fusion techniques in convolutional neural network. IEEE Access. 2021;9:161683–94. https://doi.org/10.1109/ACCESS.2021.3132062.
9. Shrinivasa S, Prabhakar C. Combining Handcrafted and Deep Features for Scene Image Classification. J Data Acquis Process. 2023;38(3):2158. https://doi.org/10.5281/zenodo.98549488.
10. Ahmed VA, Jouini K, Tuama A, Korbaa O. A Fusion Approach for Enhanced Remote Sensing Image Classification. Proceedings In (VISIGRAPP 2024). 2024;554:561. https://doi.org/10.5220/0012376600003660.
11. Tripathi S, Singh SK, Kuan LH. Bag of Visual Words (BoVW) with Deep Features–Patch Classification Model for Limited Dataset of Breast Tumours. arXiv preprint arXiv: 10701. 2022. https://doi.org/10.48550/arXiv.2202.10701.
12. Chen C, Seo H, Jun CH, Zhao Y. Pavement crack detection and classification based on fusion feature of LBP and PCA with SVM. Int J Pavement Eng. 2022;23(9):3274–83. https://doi.org/10.1080/10298436.2021.1888092.
13. Ozkaya SG, Baygin M. A New Concrete Crack Detection based on Deep Feature Extraction. IJRESM. 2023;6(6):98–105.
14. Fu R, Xu H, Wang Z, Shen L, Cao M, Liu T, *et al.* Enhanced intelligent identification of concrete cracks using multi-layered image preprocessing-aided convolutional neural networks. Sensors 2020;20(7):2021. https://doi.org/10.3390/s20072021.
15. Zhang F, Hu Z, Fu Y, Yang K, Wu Q, Feng Z. A new identification method for surface cracks from UAV images based on machine learning in coal mining areas. Remote Sensing. 2020;12(10):1571. https://doi.org/10.3390/rs12101571.
16. Bhalaji Kharthik K, Onyema EM, Mallik S, Siva Prasad B, Qin H, Selvi C, *et al.* Transfer learned deep feature based crack detection using support vector machine: a comparative study. Sci Rep. 2024;14(1):14517. https://doi.org/10.1038/s41598-024-63767-5.
17. Ahmadian V, Aval SBB, Noori M, Wang T, Altabey WA. Comparative study of a newly proposed machine learning classification to detect damage occurrence in structures. Eng Appl Artif Intell. 2024;127:107226. https://doi.org/10.1016/j.engappai.2023.107226.
18. Golding VP, Gharineiat Z, Munawar HS, Ullah F. Crack detection in concrete structures using deep learning. Sustainability. 2022;14(13):8117. https://doi.org/10.3390/su14138117.
19. Zadeh SS, Khorshidi M, Kooban F. Concrete Surface Crack Detection with Convolutional-based Deep Learning Models. IJNRCSES. 2023;10(3):25–35. https://doi.org/10.5281/zenodo.10061654.

20. Islam MM, Hossain MB, Akhtar MN, Moni MA, Hasan KF. CNN based on transfer learning models using data augmentation and transformation for detection of concrete crack. Algorithms. 2022;15(8):287. https://doi.org/10.3390/a15080287.
21. Bukhsh ZA, Jansen N, Saeed A. Damage detection using in-domain and cross-domain transfer learning. Neural Comput. Appl. 2021;33(24):16921–36. https://doi.org/10.1007/s00521-021-06279-x.
22. Zoubir H, Rguig M, El Aroussi M, Chehri A, Saadane R. Concrete Bridge Crack Image Classification Using Histograms of Oriented Gradients, Uniform Local Binary Patterns, and Kernel Principal Component Analysis. Electronics (Basel). 2022;11(20):3357. https://doi.org/10.3390/electronics11203357.
23. Zoubir H, Rguig M, Elaroussi M. Crack recognition automation in concrete bridges using Deep Convolutional Neural Networks. MATEC Web of Conferences. 2021;349:03014. https://doi.org/10.1051/matecconf/202134903014.
24. Xu H, Su X, Wang Y, Cai H, Cui K, Chen X. Automatic bridge crack detection using a convolutional neural network. Appl Sci. 2019;9(14):2867. https://doi.org/10.3390/app9142867.
25. Mundt M, Majumder S, Murali S, Panetsos P, Ramesh V. Meta-learning convolutional neural architectures for multi-target concrete defect classification with the concrete defect bridge image dataset. CVPR. 2019:11196–205. https://doi.org/10.1109/CVPR.2019.01145.
26. Lopez Droguett E, Tapia J, Yanez C, Boroschek R. Semantic segmentation model for crack images from concrete bridges for mobile devices. Proc Inst Mech Eng O. 2022;236(4):570–83. https://doi.org/10.1177/1748006X20965111.
27. Ojala T, Pietikainen M, Maenpaa T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Trans Pattern Anal Mach Intell. 2002;24(7):971–87. https://doi.org/10.1109/TPAMI.2002.1017623.
28. Yu Z, Cai R, Cui Y, Liu X, Hu Y, Kot AC. Rethinking vision transformer and masked autoencoder in multimodal face anti-spoofing. Int J Comput Vis. 2024:1–22. https://doi.org/10.1007/s11263-024-02055-1.
29. Chollet F. Xception: Deep learning with depthwise separable convolutions. CVPR. 2017:1251–8. https://doi.org/10.1109/CVPR.2017.195.
30. Marzouk MA, Elkholy M. Combining bag of visual words-based features with CNN in image classification. J Intell Inf Syst. 2024;33(1):20230054. https://doi.org/10.1515/jisys-2023-0054.
31. Hassan RQ, Sultani ZN, Dhannoon BN. Content-Based Image Retrieval System using Color Moment and Bag of Visual Words with Local Binary Pattern. KIJOMS. 2023;9(1):7. https://doi.org/10.33640/2405-609X.3274.
32. Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, *et al.* Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:04861. 2017. https://doi.org/10.48550/arXiv.1704.04861.
33. Shanshool AM, Bouchakwa M, Amous I. A Robust and Efficient Hybrid Classification Model for Early Diagnosis of Chest X-Ray Images of COVID-19. Baghdad Sci J. 2025;22(3). https://doi.org/10.21123/bsj.2024.10494.

الاستفادة من دمج السمات والشبكات العصبية الالتفافية للتنبؤ بتشققات الخرسانة

أمل عبد الباقي مريوش[1]، سعيد پاشازاده[2]، پيدرام صالِحپور[1]

[1] قسم هندسة الحاسوب، كلية الهندسة الكهربائية وهندسة الحاسوب، جامعة تبريز، تبريز، إيران.
[2] قسم تكنولوجيا المعلومات، كلية الهندسة الكهربائية وهندسة الحاسوب، جامعة تبريز، تبريز، إيران.

الخلاصة

يُعدّ الكشف عن تشققات الخرسانة أمرًا بالغ الأهمية لضمان سلامة الجسور والبنية التحتية عمومًا. تقدّم هذه الدراسة طريقة هجينة جديدة تجمع بين السمات المُستخرجة يدويًا والسمات العميقة بهدف رفع دقة التصنيف بشكل ملحوظ. تم التقاط المعلومات النسيجية والدلالية باستخدام الأنماط الثنائية المحلية (LBP)ونموذج Xception المدرَّب مسبقًا على التوالي. جرى تحويل هذه السمات باستخدام أسلوب حقيبة الكلمات البصرية (BoVW)، ثم دمجها واختيار أفضلها باستخدام خوارزمية .Apriori بعد ذلك، تم تصنيف السمات المختارة باستخدام شبكة MobileNetV3-Large خفيفة الوزن. اختُبرت الطريقة المقترحة على أربع مجموعات بيانات عامة، هي: BCD، DIMEC-Crack، CODEBRIM، وBridge ، باستخدام أسلوب التحقق المتقاطع بعشرة طيّات (10--fold cross validation) . وقد حقق النموذج المطوَّر انخفاضًا كبيرًا في معدل الخطأ ليصبح أقل من 1%. كما أظهر أداءً متميزًا من حيث مقياس الدقة (Accuracy)، إذ بلغت القيم 0.9995 و0.9983 و0.9998 و0.9993 على التوالي. وبلغت قيم الإحكام 0.9998(Precision) و1.0000 و1.0000 و0.9991، في حين سجل الاستدعاء 0.9995(Recall) و0.9938 و0.9998 و0.9986 لمجموعات البيانات نفسها. وبالمقارنة مع نماذج التعلم العميق الأخرى المدرَّبة على مجموعات البيانات ذاتها، يُظهر النموذج المقترح نتائج مشجعة وواعدة للغاية.

الكلمات المفتاحية: تنقيب قواعد الارتباط، حقيبة الكلمات البصرية (BoVW)، الشبكات العصبية الالتفافية (CNN)، دمج السمات، اختيار السمات.