Intelligent Graph Convolutional Neural Network for Road Crack Detection

Youcef Djenouri, Asma Belhadi, Essam H. Houssein, Gautam Srivastava, and Jerry Chun-Wei Lin

***Abstract*— This paper presents a novel intelligent system based on graph convolutional neural networks to study road crack detection in intelligent transportation systems. The visual features of the input images are first computed using the well-known Scale-Invariant Feature Transform (SIFT) extraction algorithm. Then, a correlation between SIFT features of similar images is analyzed and a series of graphs are generated. The graphs are trained on a graph convolutional neural network, and a hyper-optimization algorithm is developed to supervise the training process. A case study of road crack detection data is analyzed. The results show a clear superiority of the proposed framework over state-of-the-art solutions. In fact, the precision of the proposed solution exceeds 70%, while the precision of the baseline methods does not exceed 60%.**

***Index Terms*— Graph convolutional neural network, road crack detection, intelligent transportation systems, SIFT extractor.**

I. Introduction

INTELLIGENT transportation has attracted many researchers in the last five years [1], [2], [3], [4]. In particular, deep learning [5], [6] has been showing a lot of success in solving different intelligent transportation applications such as anomaly detection [7], [8], and prediction [9], [10]. One of the most important tasks of traffic units and urban planners is road maintenance, and the fundamental principles are timely detection and early warnings. The road crack detection problem aims to identify defects in roads from a large set of road images. As shown in Fig. 1, the goal of the road crack detection problem is to identify whether the road is cracked or not. In other words, the idea is to separate cracked roads from normal ones.

In [11], an encoder-decoder based model for road crack detection is proposed. The authors employed an attention mechanism to efficiently find visual features in road images. A novel approach [12], which can detect multiple spatial-frequency features, improves discrimination between high-frequency features, while requiring less computational power. According to existing literature on road crack detection, several challenges need to be overcome [11], [12], [13]. For instance, how can we build a good deep learning model with high accuracy? Are we able to explore the various dependencies between road images and use correlation for training?

Non Defect Defect

Fig. 1. An example of non road defects, and road defects.

In this work, the problems of existing road crack detection solutions are solved and a new framework combining both data correlation and a graph convolutional neural network for effective training is proposed. The main contributions of the work are listed below:

1) Develop a new algorithm for generating graphs from a set of similar images. The visual Scale-Invariant Feature Transform (SIFT) features of the input images are determined and similar images are matched. Based on image matching, a set of graphs representing similar images is generated.
2) Develop a graph convolutional neural network (GCNN) supervised by a genetic algorithm to optimize hyper-parameters of a deep learning model.
3) Evaluate the proposed system on three datasets for road crack detection. The results show clear superiority of the developed framework compared to baseline solutions.

The rest of the paper is organized as follows. Section II provides an intensive review of existing solutions for solving anomaly detection, and road crack detection problems. Section III describes the main components of the proposed framework including feature extraction, graph construction, learning process, and hyper-parameters optimization. Section IV shows experimental analysis, and finally Section V concludes the paper.

Manuscript received 31 October 2021; revised 5 July 2022 and 29 September 2022; accepted 13 October 2022. Date of publication 9 November 2022; date of current version 2 August 2023. The Associate Editor for this article was W. Wei. *(Corresponding author: Jerry Chun-Wei Lin.)*

Youcef Djenouri is with the SINTEF Digital, 0314 Oslo, Norway (e-mail: youcef.djenouri@sintef.no).

Asma Belhadi is with the Kristiania University College, 0153 Oslo, Norway (e-mail: asma.belhadi@kristiania.no).

Essam H. Houssein is with the Faculty of Computers and Information, Minia University, Minia 61519, Egypt (e-mail: essam.halim@mu.edu.eg).

Gautam Srivastava is with the Department of Math and Computer Science, Brandon University, Brandon, MB R7A 6A9, Canada, also with the Research Centre of Interneural Computing, China Medical University, Taichung 40402, Taiwan, and also with the Department of Computer Science and Math, Lebanese American University, Beirut 1102, Lebanon (e-mail: srivastavag@brandonu.ca).

Jerry Chun-Wei Lin is with the Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences, 5063 Bergen, Norway (e-mail: jerrylin@ieee.org).

Digital Object Identifier 10.1109/TITS.2022.3215538

This is the author accepted version of an article published in
IEEE Transactions on Intelligent Transportation Systems (volume: 24, issue: 8, August 2023)
https://doi.org/10.1109/TITS.2022.3215538

II. Related Work

This paper is divided into two main topics: outlier detection in intelligent transportation, and road crack detection. Recent work on both topics is discussed and analyzed below.

A. Outlier Detection in Intelligent Transportation Systems

Zhu et al. [14] developed a Convolutional Neural Network (CNN) in which traffic data observations are used to construct a database of images, each image indicating a particular state of the traffic situation in the city. Each image of traffic flow is classified into two categories: normal and abnormal case. This is done with the help of CNN so that anomalies can be identified. Huang et al. [15] studied the factors that contribute to traffic volume in an extensive metropolitan network. The obvious characteristics of an outlier are evaluated to determine whether or not they can serve as a signal of abnormal traffic behavior. To achieve this goal, a deep autoencoder learning model incorporates the spatiotemporal anomalies. The proposed technique can identify the contours around the zones that cause network anomalies, helping urban planners to accurately detect such locations. Gu et al. [16] proposed a sophisticated model for unusual passenger flows. The first step is to use a hybrid k-means and hierarchical clustering approach to identify passenger flow represented by time series data. Using a threshold technique, anomaly detection indexes are created to reflect the various outliers in passenger flow. The various observed irregularities are reported to city planners as alarms. Recent research has looked at anomaly detection in the context of maritime traffic data. To find and explain outliers in marine data, Kim et al. [17] used Shapley additive explanations with anomaly detection. In this method, attribution of characteristics of each marine observation is calculated, followed by segmentation to identify clusters of maritime observations, and each cluster is scored based on the contribution of observations to that cluster. To find anomalies in marine data, Han et al. [18] used a variational autoencoder with long-short memory. Due to the incomplete labeling of training data, the method is semi-supervised. This provides a more accurate representation of the actual situations in which sensors record typical behaviors. To capture the many local anomalies from maritime observations, Abreu et al. [19] developed a method based on visual analysis. Trip-outlier scoring is used to score the maritime trajectories and assign them an outlier score.

B. Road Crack Detection

Hacıefendioğlu et al. [20] used a deep learning-based object detection algorithm to specifically detect cracks in concrete roads under different recording, weather, and lighting conditions. As a result, existing cracks can be detected quickly and cost-effectively. Using a pre-trained Faster R-CNN, a descriptive technique for detecting cracks in images of concrete road surfaces is considered. Nguyen et al. [21] used a novel technique based on a two-stage convolutional neural network to detect defects on roads at the pixel level. It allows the context of cracks in the detected area to be determined in a second stage after removing noise or artifacts in the first stage and isolating probable cracks to a limited area. This is therefore more efficient than learning from the entire noisy original image. Fang et al. [11] present an external attention based TransUNet. It enables the transfer of detailed texture information via skip connections from shallow layers to comparable deep layers. Furthermore, the second-to-last convolutional layer of the encoding component's Transformer Block, which is endowed with the ability to explicitly model long-range dependencies of different regions within an image, improves the structural representation capability of the framework and reduces interference from shadows, noise, and other unwanted elements. Fan et al. [12] developed RAO-UNet, an encoder-decoder and residual attention module-based image frequency relationship-based network for detecting cracks in road images. RAO-UNet was able to learn many spatial frequency features compared to other approaches, improving differentiation of high frequency features while reducing the computational cost. Alfarraj et al. [22] created an Internet of Things (IoT) system using a bio-inspired deep learning approach. The proposed method involves first capturing street images with a smart mobile sensor, followed by processing by a bio-inspired self-learning algorithm. a co-evolving deep learning neural network.

C. Comparative Analysis and Discussions

From this brief literature review, we conclude that there are three types of strategies for identifying road cracks and maritime anomalies. The first group of statistical methods includes those that subject typical behaviors to a statistical process and classify remaining observations as anomalies. Approaches based on similarity fall under the second group. These methods are based on the separation of the different observations. While anomalous observations are found in sparsely populated areas, ordinary observations are found in crowded areas. Due to difficulty in capturing the distribution of normal observations and computing observations corresponding to them, statistical methods are particularly sensitive to anomalies. Similarity-based solutions used to develop non-parametric strategies have been able to deal with this problem. However, they are very sensitive to distance in computing the neighborhood. The third group consists of deep learning-based methods that use different deep architectures to address problems in the first two categories, including recurrent neural networks (RNN), convolutional neural networks (CNN), and autoencoder models. To distinguish outliers from typical behavior, they train the data and apply a binary classifier. However, all training data is considered in the learning phase. Even when using computationally intensive methods, this results in a low detection rate. These deep learning architectures also have numerous layers and a large number of hyperparameters that must be properly tuned. Accuracy can also be affected by arbitrarily adjusting these hyperparameters without performing a thorough analysis. In this paper, a hybrid fusion method is developed that combines decomposition and CNN to effectively explore maritime, and road cracks data for real-time anomaly detection. This approach is motivated by the success

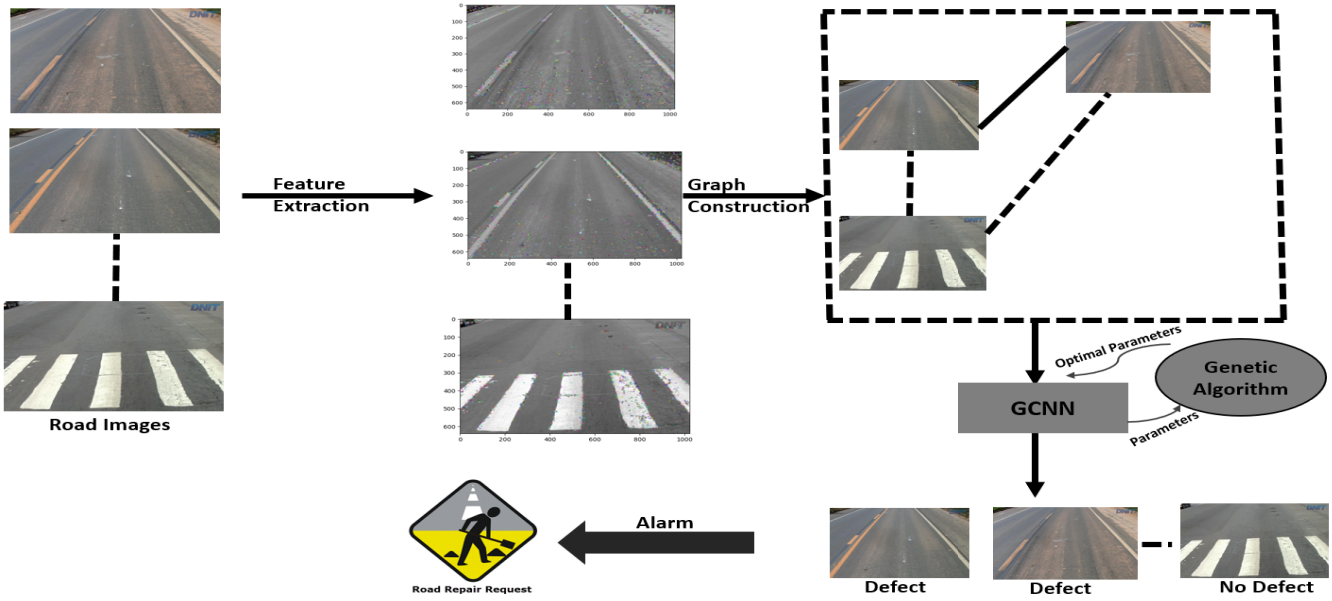

Fig. 2. IGCNN-RCD Framework Illustration: The visual features are first extracted from road images using SIFT. Then, the graph of road images is created and trained with GCNN, using the genetic algorithm to find the optimal parameters for GCNN. The results of defective roads are sent to the road repair company as an alarm.

of cluster-based algorithms [23], [24], [25] in solving complex problems.

III. IGCNN-RCD: Intelligent Graph Convolutional Neural Network for Road Crack Detection

A. Framework Overview

In this section, the IGCNN-RCD (Intelligent Graph Convolution Neural Network for Road Crack Detection) system is explained in detail, as is shown in Fig. 2. IGCNN-RCD consists of four steps:

1) **Feature Extraction:** It has the goal to extract relevant features useful in the detection process. To efficiently identify features of images, the well-known SIFT extractor algorithm is used.
2) **Graphs Construction:** After the extraction step, a graph is constructed for the learning phase. The features are used to match relevant images. The set of graphs is then created from the matching process.
3) **Learning Phase:** The generated graphs are passed to the GCNN (Graph Convolutional Neural Network) to learn different defects of road images.
4) **Hyper-parameters Optimization:** At this point, a clever method is provided to automatically determine the ideal hyperparameters for the GCNN architecture. The space of GCNN hyperparameters is explored using the evolutionary algorithm.

B. Feature Extraction

Road images with high resolution and a large number of pixels are captured by sensors and cameras in intelligent transportation environments. Depending on the image, the number of pixels can range from $250,000$ to $4,000,000$. Extracting relevant features from such images is a difficult problem in computer vision, especially for machine learning-based algorithms such as convolutional neural networks that suffer from training time for high resolution images. To efficiently extract relevant features, we developed the SIFT (Scale-Invariant Feature Transform) extractor [26]. We first consider the set of n images $F = \{F_1, F_2, \ldots, F_n\}$. SIFT is used to identify crucial frame features. The Gaussian kernel K, which serves as the foundation for the scale space function $S(F_i, \sigma)$, is used and described in Eq. 1.

$$S(F_i, \sigma) = K(F_i, \sigma) * F_i. \tag{1}$$

The spatial information of each potential key point is then determined using the interpolation procedure. The stability of the retrieved features is enabled by the generated spatially interpolated data. The interpolation function or Taylor function $Y(F_i, \sigma)$ is defined as in Eq. 2.

$$Y(F_i, \sigma) = D + \frac{dY^T}{dF_i} F_i + 0.5 F^T \frac{d^2 Y}{dF_i^2} F_i. \tag{2}$$

Next, the descriptor vector for the keypoints is created by creating different orientation histograms of the neighborhoods around 4×4 pixels. SIFT features for each I_i, say SF_i, are established after this process.

C. Graph Construction

After the feature extraction step, the set of relevant features of each image I_i, called SF_i, is determined. This step is used to prepare graphs for learning. Graphs of similar images are constructed based on matched images. First, the set of matched images is computed. Two images I_i and I_j are matched if and only if the similarity between the two feature sets SF_i and SF_j is less than a matching threshold. In this process, we used a well-known Fast Library for Approximate Nearest Neighbors

(FLANN) matching algorithm [27]. After matching, a set of graphs is generated. The list of matched images is scanned, creating an edge for each matched tuple of images (I_i, I_j). This process is repeated for all matched images. At the end, a set of graphs is generated, each representing similar images with a high matching rate between the relevant features of each image pair.

D. Learning Phase

GCNN is a deep learning architecture that uses data organized in graphs. We would like to use the convolutional layer of a CNN to work with any type of graph. Learning is based on the collection of graphs created in the previous step. The value of each node is updated based on information of its neighbors, and features of each node are updated based on the information of its neighbors. In the proposed GCNN, we can classify each node independently, the graph as a whole, the edges, or we can examine whether there is a connection between two nodes. To design an efficient GCNN, we first create the adjacency matrix A of the graph. For example, in a non-oriented graph, $A_{ij} = 1$ if and only if there is a connection between nodes i and j, and $A_{ij} = 0$ if there is no connection between i and j. We also create the node matrix H before creating the matrix shown in Eq. 3.

$$H' = \sigma\left(\widehat{D}^{-1}\hat{A}HW\right), \tag{3}$$

where σ represents a nonlinear function such as a Rectified Linear unit (ReLu), $\hat{A} = A + I$ is used to ensure that a node is connected to itself, preventing the middle node from being discarded. In addition, $\widehat{D}$ indicates a degree matrix, W is a linear layer used in the deep learning model, where this value is a learnable node-wise joint linear transformation. $\widehat{D}^{-1}$ can be considered as a normalization approach for the adjacency matrix, ensuring that features do not burst when computing the sum. This is achieved by using the mean-pooling update rule.

$$H' = \sigma\left(\widehat{D}^{-1/2}\hat{A}\widehat{D}^{-1/2}HW\right) \tag{4}$$

Then, the updating rule for the graph convolutional neural network (GCNN) is obtained. This is the graph convolutional layer currently used in most studies. In a more open version, where nodes can send any message along any edge, a node then uses a function that is invariant to permutations to aggregate all messages it has received. Let the message known as $\overrightarrow{e_{ij}}$ be the one sent from node i to node j, computed using the following message function f_e:

$$\overrightarrow{m_{ij}} = f_e\left(\overrightarrow{h_i}, \overrightarrow{h_j}, \overrightarrow{e_{ij}}\right) \tag{5}$$

After that, the following readout function is applied to each message that is received by a node in order to aggregate them:

$$f_v = \overrightarrow{h_i'} = f_v\left(\overrightarrow{h_v}, \sum_{j\in N_i} \overrightarrow{m_{ji}}\right), \tag{6}$$

where the neighbors of a node i are referred to as its N_i neighbors. This results in a message-passing neural network, or MPNN, which can in theory be used to any size graph but in fact can only handle relatively small ones. In most cases, f_e and f_v are referred to as tiny multilayer perceptrons (MLP). A more comprehensive version would be the following:

$$\overrightarrow{h_i'} = \sigma\left(\sum_{j\in N_i} \alpha_{ij} W \overrightarrow{h_j}\right), \tag{7}$$

where α_{ij} is a coefficient that is either defined explicitly which causes some shortcomings, or

$$\alpha_{ij} = \frac{\exp\left(a_{ij}\right)}{\sum_{k\in N_i} \exp\left(a_{ik}\right)}, \tag{8}$$

where,

$$a_{ij} = a\left(\overrightarrow{h_v}, \overrightarrow{h_j}, \overrightarrow{e_{lj}}\right) \tag{9}$$

Note that a is a learnable, shared, self-attention mechanism. This is called the graph attention network update rule.

E. Hyper-Parameters Optimization

Consider $\mathcal{P} = \mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{|P|}\}$, the set of all parameters used by the developed model, and $D(\mathcal{P}_i)$ is the set of possible values of $\mathcal{P}_i$. The configuration space $\mathcal{C}$ is represented by all conceivable configurations, where each configuration is a collection of the parameter values of $\mathcal{P}$. In order to find the best values for each parameter in $\mathcal{P}$, each configuration in $\mathcal{C}$ must be explored, which leads to a higher computational cost and also increases the memory requirements for certain parameters (e.g., the error rate always has the continuous values). Moreover, the total number of configurations is quite high due to the inverse relationship that exists between it and the total number of parameters and the range of values that can be assigned to each parameter, since $\prod_{i=1}^{|P|} |D(\mathcal{P}_i)|$. As the domain space of the parameter $\mathcal{P}_i$, which includes all possible values for each $\mathcal{P}_i$ in the set $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_{|P|}\}$, we denote it as $D(\mathcal{P}_i)$. Each configuration is a collection of values for all parameters in $\mathcal{P}$, and all these configurations together form the configuration space $\mathcal{C}$. To find the best values for each parameter in $\mathcal{P}$, each configuration in $\mathcal{C}$ must be examined, which consumes a lot of computational and memory resources, especially for parameters with continuous values such as the error rate. Moreover, the total number of configurations is quite high due to the inverse relationship that exists between it and the total number of parameters, as well as the range of values that can be assigned to each parameter. It is constructed such that $\prod_{i=1}^{|P|} |D(\mathcal{P}_i)|$ is displayed. For example, if we consider only $1,000$ different values for the epoch parameter (varied epoch from 1 to $1,000$), $1,000$ different values for the error rate (varied error rate from 0.001 to 1.00), and 100 different values for the number of batches (from 1 to 100), the number of all configurations in $\mathcal{C}$ is 100 million configurations. In dealing with the above use case, therefore, conventional enumeration-based methods such as Branch and Bound and A* [28] would be downright blocked. We propose an effective evolutionary computation based approach to study the configurations in $\mathcal{C}$ to solve this

problem. In this research, we will adapt the genetic algorithm to identify the best parameters for the GCNN model, which is motivated by the effectiveness of genetic algorithms in solving problems related to hyperparameter optimization [24], [29]. The following list defines the basic operations of the proposed method:

1) **Population Initialization:** The initial population, represented by the collection of individuals, is the foundation of any evolutionary computational algorithm. The potential values of each parameter in $\mathcal{P}$ define each individual. For illustration, the solution $(20, 0.75, 74)$ shows the setting where the epochs are set to 20, the error rate set to 0.75, and the batch size set to 74. The population of the proposed evolutionary computation algorithm should have a fixed size or the same number of individuals. The individuals in a population should be heterogeneous to allow better exploration of the configuration space. The initial population is produced by generating individuals while optimizing the distance between them to ensure this diversity. Therefore, the distance between the two solutions (S_1 and S_2) is defined as in Eq. 10.

$$D(S_1, S_2) = \sum_{i=1}^{|\mathcal{P}|} |S_1^i - S_2^i|, \tag{10}$$

Note that S_1^i and S_2^i are the i^{th} value of the solutions S_1, and S_2, respectively.

2) **Crossover:** This thoroughly investigates one area of the configuration space. The next crossover operator is used for each member of the current population. The crossover point is chosen randomly between 1 and $|\mathcal{P}|$, so that each individual can be divided into its left and right halves. The first child takes the left part of the first individual and the right part of the second individual, while the second child takes the right part of the first individual and the left part of the second individual. For example, if the crossover point is set to 2 and two individuals are considered, $(20, 0.75, 74)$ and $(10, 0.65, 82)$, two additional individuals are produced, the first of which is $(20, 0.75, 82)$ and the second of which is $(10, 0.65, 74)$.

3) **Mutation:** The diversification process is enabled by the mutation operator, which creates individuals outside the existing range. Each individual has a parameter that is updated and randomly selected. For example, consider this individual generated by the crossover operator $(20, 0.75, 82)$. The following individual $(15, 0.75, 12)$ is generated by the mutation operator by changing the first and third elements while the second element remains unchanged.

Each individual is first created based on the population initialization procedure and considering the heterogeneity criteria. The configuration space is then explored using crossover and mutation operators. Each individual is evaluated using the learning function as a function of detection rate to keep population size constant. Multi-objective optimization is not required at this point because road crack detection is based on a single function. The best individual is retained and the others are removed. Until the maximum number of iterations is reached, this process is repeated.

Algorithm 1 IGCNN-RCD Algorithm

1: **Input**: $I = \{I_1, I_2, \ldots, I_n\}$: the set of n road images used for the training. $I_{new} = \{I_{new}^1, I_{new}^2, \ldots, I_{new}^k\}$: the set of k new road images used for the inference.
2: **Output**: $O(I_{new})$: the output value of the crack detection of the new road images.
3: $F \leftarrow \emptyset$;
4: **for** each image $I_i \in I$ **do**
5: $\quad F \leftarrow SIFT(I)$;
6: **end for**
7: $G \leftarrow GraphConstruction(F)$;
8: $M \leftarrow HyperOptimization(GCNN(G))$;
9: $F_{new} \leftarrow \emptyset$;
10: **for** each image $I_{new}^i \in I_{new}$ **do**
11: $\quad F_{new} \leftarrow SIFT(I_{new}^i)$;
12: **end for**
13: $G_{new} \leftarrow GraphConstruction(F_{new})$;
14: $O(I_{new}) \leftarrow \emptyset$;
15: **for** $G_{new}^j \in G_{new}$ **do**
16: $\quad O(I_{new}) \leftarrow O(I_{new}) \cup M(G_{new}^j)$;
17: **end for**
18: **return** $O(I_{new})$

The pseudocode can be found in Algorithm 1. The collection of road images used for training is called the n set, while the set of new road images used for inference is called the k set (line 1). The process starts by extracting the features of each image using the SIFT extractor as explained in Section III-B (from line 3 to line 6). The graphs are extracted from the set of relevant features as explained in Section III-C (line 7). The graphs are trained in the object detection model, as explained in Section III-D (line 8). In the inference phase, the propagation of the weights of the trained model M is performed for each graph generated from fresh road photos. This step is followed by the inference step (from line 9 to line 17). We note that the training phase is a very time-consuming activity that involves the process of hyper-optimization. This phase is performed only once, regardless of the total number of images used for inference. The inference step, on the other hand, consists of only two simple loops and requires only a simple propagation of the learned model during the training phase.

IV. Performance Evaluation

Large-scale experiments were conducted to validate the developed framework. Two different types of data were used: 1) Road crack detection experiments, using three datasets (CrackTree [30], CrackForest [31], and ALE [32]) for evaluation. 2) Other use cases for urban and maritime traffic data to identify anomalies are also performed. For urban traffic data, two datasets were used: Odense[1] and Beijing,[2] and two

[1] https://www.odense.dk/
[2] https://www.beijingcitylab.com/

TABLE I

BEST PARAMETERS OF IGCNN-RCD

Dataset	Matching threshold	#Epochs	Batch Size	Learning Rate
CrackTree200	0.52	85	16	0.005
CrackForest	0.59	115	32	0.012
ALE	0.60	120	16	0.007
Odense	0.75	80	8	0.003
Beijing	0.45	82	8	0.008
Aerial Maritime	0.70	150	64	0.006
Singapore Maritime	0.50	90	32	0.008

other datasets are used for maritime data: Aerial Maritime,[3] and Singapore Maritime.[4]

The learning task used in this experiment is outlier detection. The input of IGCNN-RCD is the set of images and output is binary values indicating whether the images are outliers or not (cracked or not cracked). The goal of this experiment is to show the ability of the developed model to distinguish outliers from normal transportation images. In other words, the research question here is to what extent IGCNN-RCD can distinguish the outlier images from the normal images. To answer this research question, the evaluation is performed using different measures: Area Under Curve (AUC), Precision (P), Recall (R) and F-measure (F). Specifically, P and R are defined as follows:

$$P = \frac{TP}{TP + FP} \tag{11}$$

and,

$$R = \frac{TP}{TP + FN} \tag{12}$$

where TP represents the proportion of samples for which both the true and expected labels are positive. The term FP represents the number of samples with a positive anticipated label but a negative true label. The number of samples with positive true labels and negative anticipated labels is indicated by the symbol FN. These metrics are typical metrics for identifying the quality of outlier and road crack detection algorithms.

A. Parameter Setting

The hyperparameters of the proposed model were extracted after a large number of tests. By varying the number of generations of the genetic algorithm from 10 to 100, population size from 10 to 50, crossover rate to $\{0.5, 0.7, 0.9\}$, and mutation rate to $\{0.2, 0.4\}$, the best parameters of the proposed model were extracted and presented in Table I. These parameters are used in the rest of the experiments. In addition, Fig. 3 the results of the proposed solution obtained by varying the number of generations from 10 to 100, setting the population size to 10, the crossover rate to 0.5, and the mutation rate to 0.2.

B. Experiments on Road Cracked Data

Experiments were run with well-known benchmarks for detecting road cracks: Cracktree200, Crack Forest, and ALE,

[3]https://www.kaggle.com/ammarnassanalhajali/aerial-maritime

[4]https://www.kaggle.com/adnanenasser/singapore-maritime

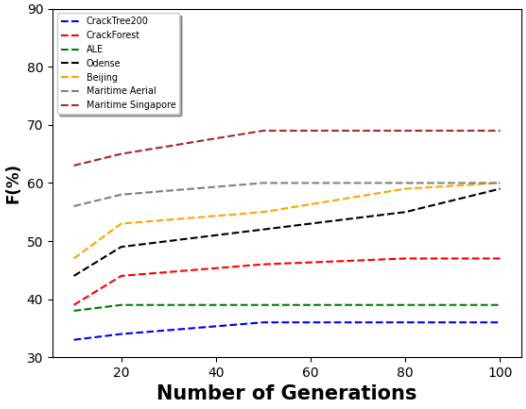

Fig. 3. Behavior of the proposed solution with different number of generations.

to verify the superiority of the proposed system. Two baseline models were used for comparison (TransUNET [11] and DELM [33]). TransUNET uses an external attention mechanism. It allows the transfer of detailed texture information from shallow layers to comparable deep layers via skip connections. DELM considers the crack detection task as a pixel-level classification and uses a U-Net based architecture to solve it. Second, the probability of occurrence of cracks and non-cracks is very different, resulting in a poorly conditioned classifier and undesirable detection performance, especially with a high false detection rate. Numerical results are shown in Table II. From this table, we can see that the three methods share certain common phenomena in the three datasets. All methods have difficulty in Precision compared to Recall and F-measure. DELM performs better than TransUNET in Recall, while TransUNET outperforms DELM in both Precision and F-measure. Although these two models benefit from their specific network architectures and optimization styles, IGCNN-RCD performs better on all measures and in all scenarios. These results are achieved thanks to the efficient combination of graph construction, GCNN training and hyperparameter optimization. Graph construction uses both SIFT features and FLANN matching to explore the correlation between similar images and create a highly connected graph of images. This enables efficient training of the GCNN network. Moreover, by exploring the GCNN parameter space, the optimal model for detecting new crack road images can be found.

C. Experiments on Urban and Maritime Traffic Data

Intensive experimental analysis was conducted to evaluate the proposed methodology using urban and maritime traffic

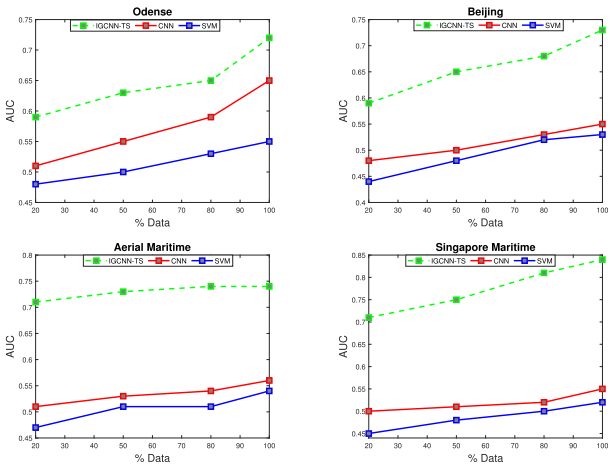

Fig. 4. Experiments on Urban, and Maritime Traffic Data.

TABLE II
RESULTS ON ROAD CRACK DETECTION

Dataset	Methods	P(%)	R(%)	F(%)
CrackTree200	TransUNet	27	46	34
	DELM	22	88	35
	IGCNN-RCD	37	94	53
CrackForest	TransUNet	40	77	53
	DELM	38	85	53
	IGCNN-RCD	48	89	62
ALE	TransUNet	52	61	56
	DELM	27	92	42
	IGCNN-RCD	55	95	70

data. We divide the data into different ranges (20%, 50%, 80% and 100%) and determine the AUC measure for each range. Two baseline algorithms are implemented. The first is a convolutional neural network [34]. The images are trained with the convolutional neural network by including feature computation and outlier determination. The second algorithm is based on traditional machine learning represented by the support vector machine (SVM) [35]. The goal is to learn the linear separation of outliers and normal images. In these experiments, we also performed hyperparameter optimization of SVM and CNN in order to provide a fair comparison with IGCNN-RCD. The results of the experiments are an average of 100 executions. As shown in Fig. 4, the results show high gain in terms of recognition rate of IGCNN-RCD compared to CNN and SVM, respectively. For example, when 20% of the images are processed, the AUC rate of all algorithms does not exceed 0.80; when the entire images are processed, the AUC of IGCNN-RCD exceeds 0.85. The success of IGCNN-RCD compared to the state-of-the-art is mainly due to the methodology used in this study. Indeed, it is a good choice to combine both feature extraction and matching to build the graphs to be trained. In this way, it is easier to find out different correlations between the set of similar images, which is very helpful for the training process. Further experiments were conducted to verify the success of the IGCNN-RCD algorithm. We chose another database based on maritime images. Maritime applications have been of great interest to many researchers recently [36], [37], [38]. We chose maritime data because the computation of features and, in particular, the generation of graphs from maritime images is very complex and computing the similarity between different maritime objects is not a simple task. Therefore, the study of maritime data is a challenging task for all deep learning architectures. We used the same AUC measure with the same algorithms, namely CNN and SVM, and changed the proportion of maritime data used from 20% to 100%. The results in Fig. 4 again show the superiority of IGCNN-RCD compared to CNN and SVM, regardless of the data used and the number of images to be processed. These interesting results underline the results found when processing urban traffic data.

V. CONCLUSION

This paper addresses outlier detection in transportation data and proposes a novel approach based on graph convolutional neural networks to solve challenging problems of existing deep learning solutions. The SIFT extractor is used to discover visual features from training images. The collection of training graphs is created using the derived features, which are also used to match similar images. Moreover, the hyperparameters of the deep learning model are optimized using the genetic algorithm and the training process is monitored. The proposed system is applied to well-known transport benchmarks. The results show that the proposed framework is useful compared to the methods used as baseline for both conventional machine learning and advanced deep learning-based methodologies. In the future, we plan to apply the proposed framework to more complex intelligent traffic data, such as trajectories and time series. We also plan to apply the proposed solutions to other intelligent transportation applications such as traffic flow prediction and connected vehicles.

REFERENCES

[1] A. Haydari and Y. Yilmaz, "Deep reinforcement learning for intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 11–32, Jan. 2020.

[2] M. B. Mollah et al., "Blockchain for the internet of vehicles towards intelligent transportation systems: A survey," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4157–4185, Mar. 2020.

[3] S. El Hamdani, N. Benamar, and M. Younis, "Pedestrian support in intelligent transportation systems: Challenges, solutions and open issues," *Transp. Res. C, Emerg. Technol.*, vol. 121, Dec. 2020, Art. no. 102856.

[4] H. Fatemidokht, M. K. Rafsanjani, B. B. Gupta, and C.-H. Hsu, "Efficient and secure routing protocol based on artificial intelligence algorithms with UAV-assisted for vehicular ad hoc networks in intelligent transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4757–4769, Jul. 2021.

[5] D. Li, L. Deng, B. B. Gupta, H. Wang, and C. Choi, "A novel CNN based security guaranteed image watermarking generation scenario for smart city applications," *Inf. Sci.*, vol. 479, pp. 432–447, Apr. 2019.

[6] S. R. Sahoo and B. B. Gupta, "Multiple features based approach for automatic fake news detection on social networks using deep learning," *Appl. Soft Comput.*, vol. 100, Mar. 2021, Art. no. 106983.

[7] K. Doshi and Y. Yilmaz, "An efficient approach for anomaly detection in traffic videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Sep. 2021, pp. 4236–4244.

[8] A. Belhadi, Y. Djenouri, G. Srivastava, A. Cano, and J. C.-W. Lin, "Hybrid group anomaly detection for sequence data: Application to trajectory data analytics," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 9346–9357, Jul. 2022.

[9] F. Zhou, Q. Yang, T. Zhong, D. Chen, and N. Zhang, "Variational graph neural networks for road traffic prediction in intelligent transportation systems," *IEEE Trans. Ind. Informat.*, vol. 17, no. 4, pp. 2802–2812, Apr. 2020.

[10] S. Yang, W. Ma, X. Pi, and S. Qian, "A deep learning approach to real-time parking occupancy prediction in transportation networks incorporating multiple spatio-temporal data sources," *Transp. Res. C, Emerg. Technol.*, vol. 107, pp. 248–265, Oct. 2019.
[11] J. Fang, C. Yang, Y. Shi, N. Wang, and Y. Zhao, "External attention based TransUNet and label expansion strategy for crack detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 19054–19063, Oct. 2022.
[12] L. Fan, H. Zhao, Y. Li, S. Li, R. Zhou, and W. Chu, "RAO-UNet: A residual attention and octave UNet for road crack detection via balance loss," *IET Intell. Transp. Syst.*, vol. 16, no. 3, pp. 332–343, Mar. 2022.
[13] K. Li, B. Wang, Y. Tian, and Z. Qi, "Fast and accurate road crack detection based on adaptive cost-sensitive loss function," *IEEE Trans. Cybern.*, early access, Sep. 21, 2021, doi: 10.1109/TCYB.2021.3103885.
[14] L. Zhu, R. Krishnan, A. Sivakumar, F. Guo, and J. W. Polak, "Traffic monitoring and anomaly detection based on simulation of Luxembourg road network," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 382–387.
[15] G.-L. Huang, K. Deng, Y. Ren, and J. Li, "Root cause analysis of traffic anomalies using uneven diffusion model," *IEEE Access*, vol. 7, pp. 16206–16216, 2019.
[16] J. Gu, Z. Jiang, W. D. Fan, J. Wu, and J. Chen, "Real-time passenger flow anomaly detection considering typical time series clustered characteristics at metro stations," *J. Transp. Eng. A, Syst.*, vol. 146, no. 4, Apr. 2020, Art. no. 04020015.
[17] D. Kim, G. Antariksa, M. P. Handayani, S. Lee, and J. Lee, "Explainable anomaly detection framework for maritime main engine sensor data," *Sensors*, vol. 21, no. 15, p. 5200, Jul. 2021.
[18] P. Han, A. L. Ellefsen, G. Li, F. T. Holmeset, and H. Zhang, "Fault detection with LSTM-based variational autoencoder for maritime components," *IEEE Sensors J.*, vol. 21, no. 19, pp. 21903–21912, Oct. 2021.
[19] F. H. O. Abreu, A. Soares, F. V. Paulovich, and S. Matwin, "Local anomaly detection in maritime traffic using visual analytics," in *Proc. EDBT/ICDT Workshops*, 2021, pp. 1–4.
[20] K. Hacıefendioğlu and H. B. Başağa, "Concrete road crack detection using deep learning-based faster R-CNN method," *Iranian J. Sci. Technol., Trans. Civil Eng.*, vol. 46, no. 2, pp. 1621–1633, Apr. 2022.
[21] N. H. T. Nguyen, S. Perry, D. Bone, H. T. Le, and T. T. Nguyen, "Two-stage convolutional neural network for road crack detection and segmentation," *Expert Syst. Appl.*, vol. 186, Dec. 2021, Art. no. 115718.
[22] O. Alfarraj, "Internet of Things with bio-inspired co-evolutionary deep-convolution neural-network approach for detecting road cracks in smart transportation," *Neural Comput. Appl.*, pp. 1–16, Oct. 2020.
[23] Y. Djenouri and J. Hjelmervik, "Hybrid decomposition convolution neural network and vocabulary forest for image retrieval," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 3064–3070.
[24] Y. Djenouri, G. Srivastava, and J. C.-W. Lin, "Fast and accurate convolution neural network for detecting manufacturing data," *IEEE Trans. Ind. Informat.*, vol. 17, no. 4, pp. 2947–2955, Apr. 2020.
[25] A. Belhadi, Y. Djenouri, J. C.-W. Lin, C. Zhang, and A. Cano, "Exploring pattern mining algorithms for hashtag retrieval problem," *IEEE Access*, vol. 8, pp. 10569–10583, 2020.
[26] S. Gupta, K. Thakur, and M. Kumar, "2D-human face recognition using sift and surf descriptors of face's feature regions," *Vis. Comput.*, vol. 37, pp. 447–456, Mar. 2020.
[27] M. Muja and D. G. Lowe, "Scalable nearest neighbor algorithms for high dimensional data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 11, pp. 2227–2240, Nov. 2014.
[28] D. Jakobovic, S. Picek, M. S. R. Martins, and M. Wagner, "Toward more efficient heuristic construction of Boolean functions," *Appl. Soft Comput.*, vol. 107, Aug. 2021, Art. no. 107327.
[29] Y. Djenouri, A. Belhadi, G. Srivastava, U. Ghosh, P. Chatterjee, and J. C.-W. Lin, "Fast and accurate deep learning framework for secure fault diagnosis in the industrial Internet of Things," *IEEE Internet Things J.*, early access, Jun. 24, 2021, doi: 10.1109/JIOT.2021.3092275.
[30] Q. Zou, Y. Cao, Q. Li, Q. Mao, and S. Wang, "CrackTree: Automatic crack detection from pavement images," *Pattern Recognit. Lett.*, vol. 33, no. 3, pp. 227–238, Feb. 2012.
[31] R. Amhaz, S. Chambon, J. Idier, and V. Baltazart, "Automatic crack detection on two-dimensional pavement images: An algorithm based on minimal path selection," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2718–2729, Oct. 2016.
[32] Y. Shi, L. Cui, Z. Qi, F. Meng, and Z. Chen, "Automatic road crack detection using random structured forests," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 12, pp. 3434–3445, Dec. 2016.
[33] J. Fang, B. Qu, and Y. Yuan, "Distribution equalization learning mechanism for road crack detection," *Neurocomputing*, vol. 424, pp. 193–204, Feb. 2021.
[34] A. R. Javed, M. Usman, S. U. Rehman, M. U. Khan, and M. S. Haghighi, "Anomaly detection in automated vehicles using multistage attention-based convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 7, pp. 4291–4300, Jul. 2021.
[35] A. Al-Dhamari, R. Sudirman, and N. H. Mahmood, "Transfer deep learning along with binary support vector machine for abnormal behavior detection," *IEEE Access*, vol. 8, pp. 61085–61095, 2020.
[36] T. Uyanık, Ç. Karatuğ, and Y. Arslanoğlu, "Machine learning approach to ship fuel consumption: A case of container vessel," *Transp. Res. D, Transp. Environ.*, vol. 84, Jul. 2020, Art. no. 102389.
[37] R. Lou, Z. Lv, S. Dang, T. Su, and X. Li, "Application of machine learning in ocean data," *Multimedia Syst.*, pp. 1–10, Feb. 2021.
[38] B. Murray and L. P. Perera, "An AIS-based deep learning framework for regional ship behavior prediction," *Rel. Eng. Syst. Saf.*, vol. 215, Nov. 2021, Art. no. 107819.

Youcef Djenouri received the Ph.D. degree in computer engineering from the University of Science and Technology (USTHB), Algiers, Algeria, in 2014. He is currently a Research Scientist with the SINTEF Digital, Oslo, Norway. He is working on topics related to artificial intelligence and data mining, with focus on association rules mining, frequent itemsets mining, parallel computing, swarm and evolutionary algorithms and pruning association rules. He has published more than 100 refereed research articles, in the areas of data mining, parallel computing, and artificial intelligence.

Asma Belhadi received the Ph.D. degree in computer engineering from the University of Science and Technology (USTHB), Algiers, Algeria, in 2016. She is a Post-Doctoral Researcher with the Kristiania University College, Oslo, Norway. She is working on topics related to artificial intelligence and data mining, with focus on smart city applications. She has published more than 50 refereed research articles in the areas of artificial intelligence and smart city applications.

Essam H. Houssein received the Ph.D. degree in computer science in 2012. He is an Associate Professor with the Faculty of Computers and Information, Minia University, Egypt. He is the Founder and the Chair of the Computing and Artificial Intelligence Research Group (CAIRG), Egypt. He has more than 100 scientific research articles published in prestigious international journals in the topics for instance meta-heuristics optimization, artificial intelligence, image processing, the IoT, and its applications. He serves as a reviewer of more than 40 journals (Elsevier, Springer, IEEE, etc.). His research interests include WSNs, the IoT, AI, bioinformatics and biomedical, image processing, data mining, and meta-heuristics optimization techniques.

Gautam Srivastava, photograph and biography not available at the time of publication.

Jerry Chun-Wei Lin received the Ph.D. degree in computer science and information engineering from National Cheng Kung University, Tainan, Taiwan, in 2010. He is currently a Full Professor with the Department of Computer Science, Electrical Engineering and Mathematical Sciences, Western Norway University of Applied Sciences (HVL), Bergen, Norway. His research interests include data mining, privacy-preserving and security, big data analytics, and social networks. He has published more than 500 research papers in peer-reviewed international conferences and journals, which have received more than 1900 citations. He is the Co-Leader of the Popular SPMF Open-Source Data-Mining Library and the Editor-in-Chief of *Data Mining and Pattern Recognition* (DSPR), an Associate Editor of several IEEE journals including IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, and IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING.