IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 9, 2020, accepted September 26, 2020, date of publication October 2, 2020, date of current version October 15, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3028395

Reliable and Rapid Traffic Congestion Detection Approach Based on Deep Residual Learning and Motion Trajectories

MOHAMED A. ABDELWAHAB[1,2], MOHAMED ABDEL-NASSER[3,4], AND MAIYA HORI[5], (Member, IEEE)

[1]Center of Coevolutionary Research for Sustainable Communities, Kyushu University, Fukuoka 819-0395, Japan
[2]Department of Electrical Engineering, Faculty of Energy Engineering, Aswan University, Aswan 81542, Egypt
[3]Department of Computer Engineering and Mathematics, Rovira i Virgili University, 43007 Tarragona, Spain
[4]Department of Electrical Engineering, Faculty of Engineering, Aswan University, Aswan 81542, Egypt
[5]Platform of Inter/Transdisciplinary Energy Research, Kyushu University, Fukuoka 819-0395, Japan

Corresponding authors: Mohamed A. Abdelwahab (abdelwahab@aswu.edu.eg) and Mohamed Abdel-Nasser (mohamed.abdelnasser@urv.cat)

ABSTRACT Traffic congestion detection systems help manage traffic in crowded cities by analyzing videos of vehicles. Existing systems largely depend on texture and motion features. Such systems face several challenges, including illumination changes caused by variations in weather conditions, complexity of scenes, vehicle occlusion, and the ambiguity of stopped vehicles. To overcome these issues, this article proposes a rapid and reliable traffic congestion detection method based on the modeling of video dynamics using deep residual learning and motion trajectories. The proposed method efficiently uses both motion and deep texture features to overcome the limitations of existing methods. Unlike other methods that simply extract texture features from a single frame, we use an efficient representation learning method to capture the latent structures in traffic videos by modeling the evolution of texture features. This representation yields a noticeable improvement in detection results under various weather conditions. Regarding motion features, we propose an algorithm to distinguish stopped vehicles and background objects, whereas most existing motion-based approaches fail to address this issue. Both types of obtained features are used to construct an ensemble classification model based on the support vector machine algorithm. Two benchmark datasets are considered to demonstrate the robustness of the proposed method: the UCSD dataset and NU1 video dataset. The proposed method achieves competitive results (97.64% accuracy) when compared to state-of-the-art methods.

INDEX TERMS Congestion, deep learning, residual network, traffic surveillance system.

I. INTRODUCTION

Large cities contain millions of people that use different means of transportation, including buses, taxies, motorcycles, and bicycles. The transportation infrastructure in large cities cannot accommodate continuous growth in the number of vehicles, which leads to traffic congestion. Additional sources of traffic congestion are weather conditions, traffic demand, and traffic-influencing events, such as accidents [1]. Traffic gridlock negatively affects the quality of life, career opportunities, and safety of people. Additionally, stopping and starting vehicles in traffic gridlock consumes more fuel than normal traffic operations, leading to additional air pollution. Therefore, transport policy makers in several large cities have exploited intelligent traffic management systems (TMSs) to mitigate and prevent traffic congestion and improve overall traffic efficiency. TMSs include automated traffic monitoring systems that analyze images/videos captured by closed-circuit television cameras [2] to detect the status of traffic (e.g., light, medium, or heavy) and measure traffic flow.

Traffic congestion classification approaches can be categorized according to the features they utilize as motion-based [3] and texture-based methods [4], [5].

The associate editor coordinating the review of this manuscript and approving it for publication was Huiyu Zhou.

However, the employment of either motion or texture features alone in traffic congestion detection systems has a few shortcomings. For example, in the case of motion-based traffic congestion classification methods, stopped vehicles may not be tracked, meaning a heavy congestion traffic scene may be misclassified as an empty road. In the case of texture-based methods, a few vehicles in a scene may be classified as having low traffic congestion, even if they exhibit slow motion. Additionally, most existing texture-based traffic classification methods extract texture features from single frames and do not model the evolution of textures in traffic videos. Fig. 1 presents examples from a traffic video dataset provided by the University of California San Diego (UCSD videos dataset) [6] that contains three different traffic congestion classes (light, medium, and heavy). In this figure, each video is represented by two different frames. It is evident that not only do the number of vehicles between the two frames change, but there are also differences in vehicle motion patterns (i.e., vehicle displacement between the two frames). Additionally, all three videos have different illumination conditions.

(a) Light (b) Medium (c) Heavy

FIGURE 1. Examples of different traffic congestion statuses, weather conditions, and illumination changes.

A reliable traffic congestion detection method should rapidly provide accurate results under the presence of various illumination conditions and traffic events. To fulfill these requirements, this study proposes a reliable and rapid traffic congestion classification method based on the modeling of video dynamics using a deep residual network and motion trajectories. The proposed method consists of three main steps: (1) modeling the evolution of texture features in videos, (2) extracting motion trajectories, and (3) constructing an ensemble classifier using the support vector machine (SVM) algorithm. To classify traffic congestion rapidly, the proposed method handles frames in batches instead of analyzing an entire video. In step (1), both deep convolutional neural networks (CNNs) and handcrafted texture analysis methods are assessed and compared. Existing methods either extract features from a single frame or from an entire video. Few studies have considered modeling the temporal information within traffic videos. To enrich the extracted texture features, we model the evolution of the texture features in each batch of frames using a representation learning method called learning-to-rank (LTR). The use of LTR helps capture the latent structures in traffic videos and exploit the temporal information embedded in traffic videos, which can significantly improve detection accuracy congestion.

It should be noted that the powerful representational ability of deep residual networks and the adopted representation learning method yield a significant improvement in the accuracy of classification results. In step (2), sparse corner points in each batch of frames are extracted and tracked to obtain motion trajectory features. Because the extracted corner points lie in both vehicle and background regions, existing methods use various algorithms to suppress corner points in background regions. However, corner points in regions of stopped vehicles are also suppressed because they are considered as background corner points, which yields inaccurate results. To address this issue, we propose an algorithm for removing the corner points in background regions while retaining the points corresponding to stopped vehicles. Ultimately, the features obtained in steps (1) and (2) are employed to construct an ensemble classifier based on an SVM to classify traffic congestion into three classes of light, medium, and heavy.

The key contributions of this article can be summarized as follows:

- We propose a rapid and reliable traffic congestion detection method that can overcome the limitations of existing approaches.
- We present a feasible method for modeling the dynamics of traffic videos based on representation learning and deep residual learning.
- We propose an algorithm for distinguishing the motion features of stopped vehicles from background objects. Most existing motion-based traffic congestion detection approaches do not thoroughly address this issue.
- We present comprehensive analysis of the performance of different deep learning models and handcrafted texture analysis methods for traffic congestion detection.
- We achieve the state-of-the-art traffic congestion detection results on two benchmark datasets with a processing time of less than 16 ms.

The remainder of this article is organized as follows. Section 2 discusses related methods. Section 3 presents the proposed method. Section 4 presents the experimental results and discussion. Section 5 concludes this article and discusses future research directions.

II. RELATED WORK

In general, traffic congestion detection methods can be classified into vehicle-based or holistic methods. Here, we discuss the strengths and weaknesses of both categories.

Vehicle-based methods: The typical pipeline of a vehicle-based method consists of vehicle detection, tracking, extraction of trajectories, and training of a classifier to distinguish traffic congestion classes (light, medium, or heavy) [7]–[9].

In a study performed by Kim *et al.*, the differences between successive frames and various adaptive thresholding methods were used to detect moving objects [7]. Wavelet coefficients were extracted from moving objects and then fed into a neural network to detect and track vehicles. In the training phase described in [9], a codebook was generated based on scale-invariant feature transforms (SIFTs) [10]. In the detection stage, vehicles were detected using the generated codebook.

Vehicle occlusion is one of the major challenges faced by visual traffic congestion detection methods. To address this issue, Huang and Barth [8] proposed two distinct solutions for low and high occlusion. For low occlusion, local features were adopted in the vehicle detection and tracking steps. For high occlusion, color probabilities and local features were used to detach occlusion areas from a scene and track each vehicle individually.

Additionally, the numbers of vehicles moving on a road have been utilized in many automated methods as indicators of traffic flow. These approaches assume that the number of vehicles passing a predefined line per minute represents the traffic flow. In such approaches, vehicles are detected, tracked, and then counted. For example, the low-rank decomposition method was applied to input video frames in [11] to locate low-rank components that describe background objects and spare outliers representing foreground objects (vehicles moving on roads and noise). The extracted vehicles were then tracked using a Kalman filter.

Deep neural networks (DNNs) have been extensively employed in numerous artificial intelligence applications, including object detection, image segmentation, and classification [12]. Singh and Jain [13] employed various deep-learning-based object detection models to achieve accurate counts of vehicles and analyze traffic patterns. They tested a region-based CNN (R-CNN), fast R-CNN, and faster R-CNN. The fast R-CNN and faster R-CNN models are modified versions of the R-CNN with lower computational costs. The faster R-CNN replaces the selective search used in the R-CNN and fast R-CNN with a region proposal network. Abdelwahab [14] proposed an efficient and fast method for vehicle counting without employing any vehicle-tracking algorithms. To reduce processing time, the author proposed creating a background model for restricted regions of interest (ROIs) in video frames instead of using entire frames. Moving vehicles were recognized as foreground objects when they passed through a narrow ROI. Then, the foreground objects were used to calculate the number of vehicles. The vehicle-based methods discussed above rely heavily on the efficiency of the employed vehicle detection and tracking algorithms. This dependency can become a hindrance in the case of crowded scenes or poor weather conditions, where vehicle detection and segmentation are challenging tasks.

Holistic approaches: Unlike vehicle-based methods, holistic approaches use spatio-temporal features extracted from videos to produce global representations of a particular scene instead of handling individual vehicles [3]–[5], [15], [16]. Porikli and Li [15] exploited the motion vectors and discrete cosine transform coefficients embedded in the MPEG compressed domain as spatio-temporal features. The main shortcoming of this method is that it can only be applied to MPEG videos. Asmaa *et al.* [16] proposed microscopic and macroscopic methods for estimating traffic density. In their microscopic method, three motion features of traffic flow, velocity, and the rate of road occupancy were computed after detecting and tracking vehicles. In their macroscopic method, video frames were split into small blocks. For each frame, a block-search algorithm was used to obtain the best-matched block in the subsequent frames. After obtaining motion vectors, two macroscopic features called mean flow velocity and density were calculated. This macroscopic method yielded the highest accuracy. However, the computational cost of the block-matching algorithm is very high, making this method unsuitable for real-time traffic monitoring.

Riaz and Khan [3] employed motion features obtained by tracking sparse corners in traffic videos to describe traffic flows. Before computing motion features, motion vectors containing low displacements or directions extracted from the main directions of vehicles were excluded. Four motion features were computed for each video: the mean and standard deviation of velocity values, average length of motion vectors, and number of motion vectors. The main shortcomings of this method are twofold. First, the mean velocity of the motion vectors is calculated for an entire video without considering variations in traffic flow. Second, this method excludes the motion vectors of stopped vehicles, resulting in inaccurate traffic congestion estimates.

Several recent studies, including [4], [5], [17], [18], have employed DNNs for traffic monitoring. Luo *et al.* [4] used texture features to classify traffic congestion in videos without considering motion information. Two different texture analysis methods were evaluated: codebook descriptors and texture features extracted by CNNs. In the case of the codebook-based features, SIFT features were used to compute four visual codebook descriptors: vectors of locally aggregated descriptors, improved Fisher vectors, locality-constrained linear coding, and bags of visual words. In the case of the CNN-based features, the last fully connected layer in a pre-trained CNN was used to generate a feature vector with a length of 4096 for each frame in an input video.

Three traffic congestion classification approaches based on deep CNN models were proposed in [5]. In the first approach, a CNN was employed as a feature extractor and a support vector regression algorithm was used to map the traffic congestion of a scene to a value in [0, 1]. In the second approach, a CNN network was used to classify each pixel into three semantic classes: background, road, and vehicle. To that end, a patch was extracted around each pixel, meaning the computational complexity of this approach is extremely high. In the third approach, patch classification was used instead of pixel-wise classification. In each patch, the percentages of background, road, and vehicles were calculated

as regression values. This approach also has a high computational cost because it considers hundreds of patches per frame. Additionally, these three approaches were used for still images without considering the temporal (motion) information in traffic videos. Wang *et al.* [19] fine-tuned a pre-trained deep residual neural network to detect traffic congestion. They constructed a traffic image dataset to validate their model for different illumination conditions, weather conditions, and traffic scenarios. Based on this regional dataset, they achieved an accuracy of 95%. Sun *et al.* [20] proposed a systematic method to classify congestion based on an attention module and deep supervised inception network. For a large dataset of low-frame-rate videos gathered from a traffic surveillance system, they achieved an accuracy of 95.77%. Zhang *et al.* [21] proposed a congestion prediction model using CNNs and a long short-term memory neural network. Raw snapshots of traffic congestion maps were represented as a series of matrices to train their model. This method does not directly consider vehicles or the visual characteristics of roads. Therefore, it may not provide a realistic estimation of traffic congestion.

As discussed above, existing traffic congestion classification methods tend to achieve poor results in the presence of illumination changes, poor weather conditions, and various traffic events, such as accidents and stopped vehicles. Although some existing methods yield promising results, their computational costs are very high (e.g., [5]). The main goal of this study was to develop a reliable and rapid traffic congestion classification method that can provide accurate results under the aforementioned conditions with a very short execution time (i.e., low computational cost). To this end, this article proposes a novel traffic congestion detection method that models video dynamics using a deep residual network and motion trajectories.

III. PROPOSED METHOD

Fig. 2 presents the framework of the proposed traffic congestion classification method. In the training phase of the proposed method, we train two SVM models called SVM-1 and SVM-2 that classify the traffic congestion in each input video as low, medium, or heavy. Based on training videos and the corresponding ground-truth (GT) results (actual class label of each video), we extract texture features from N frames and use the LTR method to generate a compact feature representation to train SVM-1. We also extract motion features from training videos in parallel and combine them to train SVM-2. After training SVM-1 and SVM-2, we aggregate their results to derive the final classification results for traffic congestion in input videos. In the testing phase, both texture and motion features are extracted from test videos and fed into SVM-1 and SVM-2 to obtain traffic congestion prediction scores. An aggregation function is used to determine the final traffic congestion class. The proposed method provides traffic congestion classes for every N frames of an input video.

Because the main objective of the proposed method is to achieve reliable traffic congestion classification results,

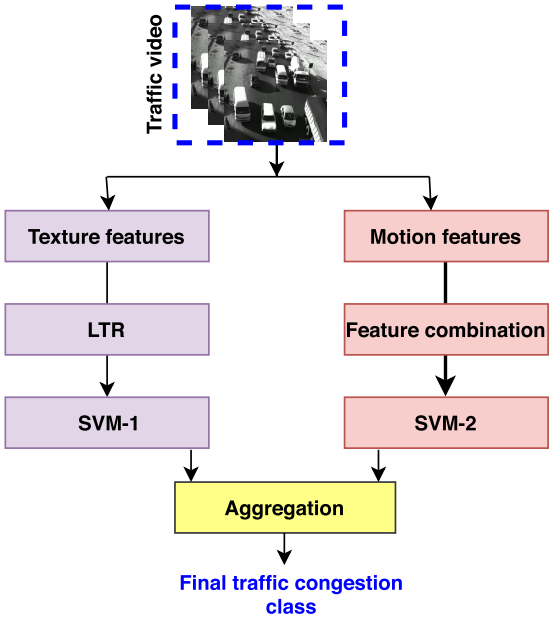

FIGURE 2. Framework of the proposed method.

we propose the use of robust texture and motion features extracted from traffic videos. Unlike related methods that extract global features from each frame, the proposed approach extracts local motion and texture features from batches of frames. To this end, each frame is split into $m \times n$ blocks, a feature vector f_i is extracted from each block, and then all feature vectors are concatenated into a single feature vector. Extracting local features can reduce the effects of illumination changes and the ambiguity of certain videos caused by poor weather conditions.

Our secondary objective is to achieve a very short processing time such that the proposed approach can be integrated with real-time applications. Consequently, congestion classification results are calculated for batches of N frames, where N can be selected according to the framerate of an input video. After obtaining the feature vectors for a batch of frames, the LTR technique is applied to the N texture features to produce a single feature vector that models the temporal evolution of traffic congestion. SVM models are trained separately using texture and motion features. In the testing phase, the scores of both models are aggregated to obtain final predictions for traffic statuses. We detail each step of the proposed approach below.

A. TEXTURE FEATURES

Texture features (appearance features) are used to analyze traffic congestion. To select a texture analysis method that properly represents traffic congestion in videos, both deep CNN models and handcrafted texture analysis methods were evaluated and compared. For each texture analysis method, a vector of length $1 \times L$ was computed for each frame.

The features of every batch of N frames ($N \times L$) are fed into the LTR method [22], which produces a compact and descriptive feature vector with a size of $1 \times L$. It should be noted that

the used LTR method does not require ground-truth results to learn compact representations for batches of frames (unsupervised method). The obtained compact feature vectors are inputted into an SVM classifier to differentiate between three congestion classes: *light*, *medium*, and *heavy*. The texture features employed in our experiments can be categorized as handcrafted and deep CNN features. We describe these features in detail below.

1) DEEP-CNN-BASED FEATURES

Unlike handcrafted features, which are sensitive to illumination changes and noise, a deep CNN can generate robust texture features for each frame without manual supervision. In this study, several pre-trained CNN models were considered for feature extraction, namely VGG19 [23], GoogleNet [24], inceptionv3 [25], and ResNet101 [26]. VGG19 [23] contains a total of 47 layers with several successive convolution layers, and each layer is followed by a rectified linear unit layer. The architecture of GoogleNet [24] consists of several paths with 22 weight layers—meaning processing is conducted in parallel, rather than sequentially. The basic block in GoogleNet is the "inception module," where the processing of many convolution kernels is performed in parallel. Inceptionv3 [25] has a small number of parameters and excellent computational efficiency. Therefore, it can facilitate complexity similar to that of VGGNet, but with more deep layers. It should be noted that if the depths of the VGG19, GoogleNet, and inceptionv3 models are increased, accuracy becomes saturated and then decreases. Unlike these methods, in ResNet, the input for one layer is passed directly or through skip connections to another layer (called identity mapping). The skip connections employed in ResNet help enhance the performance of CNNs with a large number of layers. Additionally, ResNet can reduce the effects of the vanishing gradient problem. Additional details can be found in [26]. In recent years, ResNet has enhanced the performance of various computer vision applications, including semantic segmentation, object detection, and image classification, based on the powerful representational ability of deep residual networks.

All pre-trained models considered in our experiments are available in Matlab2019a. These models were originally trained to classify images with dimensions of 224×224 pixels. However, they can be exploited as feature extractors for images with arbitrary dimensions. For each CNN model, texture features were extracted from the final layer with a dimension of $1 \times L$ for each frame, where L depends on the CNN model.

2) HANDCRAFTED TEXTURE FEATURES

Three commonly used texture analysis methods were considered in this study: histogram of oriented gradients (HOG) [27], local binary patterns (LBP) [28], and local directional number patterns (LDN) [29]. Handcrafted features are extracted locally instead of calculating global features for an entire frame. For texture analysis methods, an input frame is split into $m \times n$ blocks and a feature vector is extracted from each block. The final texture vector, which has a length of L ($L = m \times n$), is constructed by concatenating the extracted vectors from all blocks.

3) LEARNING A COMPACT REPRESENTATION

The LTR method was adopted to generate a compact description of texture features in each batch of frames [30], [31]. Assume that we have a sequence of frames, where each frame at time t is described by a vector $F_t \in \mathbb{R}^D$. In this case, the entire sequence can be represented by $X = [F_1, F_2, \ldots, F_N]$, where N is the number of frames. The LTR method is used to model the relative ranks of frames (i.e., F_2 comes after F_1, which can be written as $F_2 \succ F_1$). LTR learns the orders of smoothed versions of feature vectors to alleviate the effects of sudden changes in extracted texture features caused by illumination changes. We define a sequence $Z = [z_1, z_2, \ldots, z_N]$, where z_t is the result of processing the feature vectors from time 1 to time t ($F_1 \rightarrow F_t$) using the time-varying mean method as follows:

$$\delta_t = \frac{1}{t}\sum_{i=1}^{t} F_i. \tag{1}$$

The next formula is utilized to produce z_t, which is the normalized version of δ_t.

$$z_t = \frac{\delta_t}{\|\delta_t\|}. \tag{2}$$

Given the smoothed vectors, LTR learns their ranks (i.e., $z_n \succ z_{n-1} \succ z_{n-2} \succ \cdots \succ z_1$) and linear LTR learns pairwise linear functions $\tau(z_t; \chi)$, where $\chi \in \mathbb{R}^D$. The ranking score of z_t is computed as $\tau(z_t; \chi) = \chi^T.z_t$. The LTR method optimizes the parameters χ of $\tau(z_t; \chi)$ using the objective function in Equation (3) with the constraint $\forall t_i, t_j \quad z_{t_i} \succ z_{t_j} \iff \chi^T.z_{t_i} \succ \chi^T, z_{t_j}$ as follows [22]:

$$\begin{aligned} \arg\min_{\chi} \quad & \frac{1}{2}\|\chi\|^2 + \kappa \sum_{\forall i,j,z_i \succ z_j} \epsilon_{ij}, \\ s.t. \quad & \chi^T(z_{t_i} - z_{t_j}) \geq 1 - \epsilon_{ij}, \\ & \epsilon_{ij} \geq 0, \end{aligned} \tag{3}$$

where κ is the regularization parameter and ϵ is the margin of tolerance. The constraint of the objective function encourages the ranking score (prediction score) of the difference between the feature vectors of two infrared images z_{t_i} and z_{t_j} acquired at times t_i and t_j, respectively, to be greater than $1 - \epsilon_{ij}$. The margin of tolerance should be greater than or equal to zero. In other words, if z_{t_i} occurs after z_{t_j}, then the ranking score of z_{t_i} should be greater than that of z_{t_j}. In our model, traffic congestion evolution information is encoded in the parameters χ, which can be viewed as the principled, data-driven, temporal pooling of the evolution of traffic congestion in videos. Therefore, the parameters χ are used in our method to describe the evolution of traffic.

B. MOTION TRAJECTORIES

Fig. 3 illustrates the steps for extracting motion features from traffic videos. We extract corner points from the first frame, use the Kanade-Lucas-Tomasi tracker [32] to track such points in subsequent frames, and eventually obtain point trajectories. The average velocity of moving vehicles is computed for every N frames to realize rapid traffic congestion classification. It should be noted that motion features are dependent on pixel coordinate displacement. Therefore, distant vehicles appear to move very slowly and close vehicles appear to move very fast in a 2-D image plane. To reduce the effects of this discrepancy, we extract motion features from a fixed ROI, as shown in Fig. 4. Within an ROI, the differences between the speeds of vehicles are negligible.

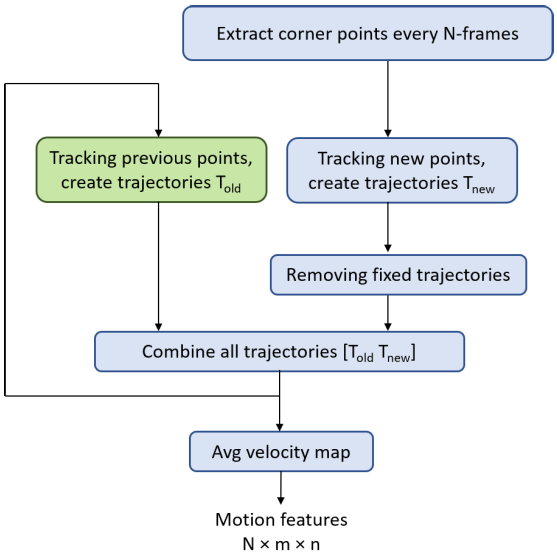

FIGURE 3. Extracting motion features while maintaining the trajectories of stopped vehicles.

FIGURE 4. Extracting motion features from an ROI.

The number of moving vehicles varies over time based on the entry of new vehicles and departure of old vehicles. For this reason, new corner points are extracted every N frames [33], and the new points and previous points are both tracked to obtain trajectories T_{new} and T_{old}, respectively. The trajectories of fixed corner points, such as those on background objects, or corner points with low displacement, such as those on trees, are discarded. To detect such fixed corner points, the overall displacement for all points is calculated over N frames. Then, the corner points with path lengths less than a predefined threshold (trajectory length < 2 pixels over five frames) are excluded.

1) PRESERVING THE FEATURES OF STOPPED VEHICLES

To prevent excluding corner points corresponding to stopped vehicles, we only filter new corner point trajectories T_{new} from the fixed points (Fig 3). Therefore, the previous trajectories T_{old} (corner points tracked in the previous N frames) are not filtered because these trajectories represent moving vehicles in the previous N frames and one or more vehicles may have stopped in the current N frames. Therefore, we can overcome one of the main drawbacks of most motion-based traffic congestion classification methods, where stopped vehicles are typically classified as empty roads.

Fig. 5 presents frame number 7871 of the Nile University traffic video (NU1) [34]. We present the corner points before and after removing fixed points. In this example, we focus on two regions: one region contains a stopped vehicle that was moving in the previous batch of frames and has stopped, and the other region is part of the background. One can see that the proposed background-point-removal technique retains the corner points of the stopped vehicle while eliminating the other fixed corner points.

FIGURE 5. Example of excluding the corner points in the background while retaining the corner points of stopped vehicles. The top frame presents all corner points, and the bottom frame presents the corner points after excluding the points in the background.

2) EXTRACTING MOTION FEATURES

To construct a motion feature vector, the displacement magnitude s_i for each corner point between consecutive frames can be computed easily as

$$s_i = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}, \tag{4}$$

where (x_1, y_1) and (x_2, y_2) are the corner point coordinates in two consecutive frames. To mitigate errors in point tracking,

we exclude corner points with displacements greater than a predefined threshold S_{max} (very high displacements). In this study, S_{max} was set to 10.

As explained previously, features are extracted locally for each block, meaning each frame is divided into $m \times n$ blocks. The average displacement μ_k for all corner points in block k over N frames is calculated as follows:

$$\mu_k = \frac{1}{MN} \sum_{i=1}^{N} \sum_{j=1}^{M} s_{ij}, \tag{5}$$

where M is the number of corner points in the block and N is the number of frames. If a block k does not contain any corner points corresponding to vehicles, meaning it represents an empty road, then μ_k is set to a high value ($2 \times S_{max}$). The goal of setting this value is to consider blocks that represent very light traffic and distinguish them from other blocks that represent extremely heavy (i.e., jammed) traffic, which has $\mu_k = 0$. After obtaining the μ_k values for all blocks, we concatenate them into a single feature vector for N frames as

$$\mu = [\mu_1, \mu_2, \ldots, \mu_{mn}]. \tag{6}$$

In the training phase, the motion feature vectors $(\mu)_s$ of the training videos and their traffic class labels are fed into an SVM model, as shown in Fig. 2 (right).

C. AGGREGATING THE PREDICTIONS OF CLASSIFIERS

In the literature, several traffic congestion detection methods have used SVMs and achieved good results (e.g., [4], [35]). SVMs can be used for both linearly separable and nonlinearly separable data based on a kernel trick [36]. Therefore, we employed this trick in our model. An SVM classifier identifies a hyperplane as a decision surface to discriminate between positive and negative instances according to a maximum margin. For a labeled training set of the form (x_i, y_i), $i = 1, 2, \ldots, k$, $x_i \epsilon R^n$ are the feature values of instance i, $y_i \epsilon[-1, 1]$ is the label of the current sample, n is the number of features, and k is the number of instances. An SVM classifier solves the following optimization problem:

$$\|\omega\|_\omega^2 + C \sum_{i=1}^{k} \xi_i \; where \; y_i(\omega^T(xi) + b0) \geq (1 - \xi_i), \; \xi_i \geq 0. \tag{7}$$

As indicated in the expression of SVM optimization, the soft margin parameter C can be used to determine the size of the margin required to avoid misclassifying each training image. The weight vector ω is normal to the separating hyperplane, the parameter ξ is the degree of flexibility of the algorithm for fitting the data, and $b0$ is the bias. In a nonlinear SVM, training data are mapped into a higher-dimensional space using a kernel function as follows:

$$K(x_i, x_j) = (\Phi^T(xi).\Phi(x_j)). \tag{8}$$

In this study, we adopted the linear kernel $K(x_i, x_j) = x_i.x_j$ for the texture features and the radial basis function (RBF) kernel for the motion features. The RBF can be expressed as follows:

$$K(x_i, x_j) = exp - (\gamma \parallel x_i - x_j \parallel_2^2), \tag{9}$$

where γ=1/2σ^2, $\|x_i\text{-}x_j\|_2^2$ computes the squared Euclidean distance between any two feature vectors x_i and x_j, and σ is a control parameter for the RBF.

The goal of the proposed hybrid classifier is to integrate the descriptive power of motion and texture features to obtain reliable congestion classification results. To determine the congestion class of a batch of N frames, separate SVM classifiers are trained for texture and motion features. Given the posterior probabilities of the two classifiers ρ_i, $i = 1, 2, \ldots, U$, the N input frames, denoted as Q, are labelled with a class α_i (light, medium, or heavy) using a sum rule as follows:

$$\begin{aligned} \text{assign } Q \rightarrow \alpha_j \quad & \text{if } (1-U)P\left(\alpha_j\right) + \sum_{i=1}^{U} P\left(\alpha_j|\rho_i\right) \\ & = \max_{k=1}^{q} \left[(1-U)P\left(\alpha_k\right) + \sum_{i=1}^{U} P\left(\alpha_k|\rho_i\right) \right], \end{aligned} \tag{10}$$

where $P\left(\alpha_j\right)$ is the a posteriori probability of α_j, U is the number of classifiers, and q is the number of classes. In our method, $U = 2$ and $q = 3$.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

Most publicly available traffic classification datasets consist of still images, and there are only a few traffic congestion datasets containing videos and corresponding ground-truth labels. We did not consider still image datasets for evaluating our method because it depends on both motion trajectories and texture features. Two benchmark traffic congestion detection datasets were used in this study: the UCSD dataset [6] and the long NU1 video [34]. First, the UCSD dataset was considered to compare the performance of the proposed method to those of state-of-the-art traffic congestion classification methods. The NU1 video (45 minutes) was considered to assess the performance of the proposed method on long videos of poor quality.

A. EXPERIMENTS ON THE UCSD DATASET

In this section, the UCSD dataset (254 videos) [6] is used to evaluate the proposed method. The videos in this dataset were recorded on a highway during the day under various conditions, including clear, overcast, and rainy weather.

Accuracy of texture features: We begin by classifying traffic congestion based solely on texture features. To select the most suitable texture features, various texture analysis methods are exploited, including handcrafted features (HOG, LBP, and LDN), and CNN-based features (VGG19, GoogleNet, Inceptionv3, and ResNet101). The number of frames was not equal for all videos, so the first 40 frames were extracted from each video. In this experiment, for the sake of comparison to related methods, offline classification

TABLE 1. Different texture features and corresponding accuracy values for individual and combined frames exploiting the LTR method.

Method	Length	Accuracy %					
		Individual frames			LTR		
		Frame 11	Frame 21	Frame 31	30-frames	10-frames	5-frames
HOG	12852	92.81	92.91	92.13	71.26	80.71	90.94
LBP	531	94.49	92.52	93.70	73.23	89.76	94.88
LDN	504	90.94	94.49	92.13	68.90	88.58	92.52
VGG19	4096	90.94	94.49	92.13	80.31	88.19	94.09
GoogleNet	1024	92.52	89.37	90.94	74.80	84.25	90.55
Inceptionv3	2048	94.49	88.98	94.49	70.47	84.25	89.76
ResNet101	2048	92.52	93.31	93.70	73.41	87.57	**95.28**

was conducted, meaning a classification decision was only computed once for each video.

In this study, we compared two approaches: i) extracting texture features from individual frames and ii) extracting texture features from a batch of frames (N frames). First, texture features were extracted from individual frames, such as frame 11, frame 21, or frame 31, in each video. Next, the extracted feature vectors were utilized to represent the entire video sequence in the training and testing phases. Second, we used the LTR method to generate compact representations of the features extracted from N frames for each video. Different numbers of frames (N) were tested (30, 10, and 5 frames). In both cases, after extracting texture features, an SVM model was trained based on the extracted features. In the testing phase, the trained model was used to classify test videos into light, medium, and heavy categories. For the sake of fair comparison, we used the same evaluation criteria discussed in [6] based on four-fold cross validation. Additionally, the same video indices were used in our experiments.

Table 1 presents the obtained accuracy values for the analyzed texture analysis methods. One can see that the accuracies obtained for the individual frames vary from one frame to another. For example, VGG19 obtains its best accuracy (94.49%) for Frame 21, whereas ResNet101 achieves its best accuracy (93.70%) for Frame 31. In the case of the LTR method, we studied the tradeoff between the number of frames considered for congestion classification and accuracy. In Table 1, we present the classification accuracy results for traffic congestion classes (heavy, medium, and light) based on 5, 10, and 30 frames using the LTR method. One can see that the use of 30 frames with ResNet101 yields an accuracy of 73.41% while the use of 10 frames yields an accuracy of 87.57%. When we classify traffic congestion based on batches of five frames, we achieve the highest accuracy (95.28%) using texture features.

A confusion matrix is presented in Table 2. One can see that five heavy traffic videos are misclassified as medium traffic videos. In our experiments, we evaluated the performance of the compact representations generated by forward LTR (LTR_{forw}) and reverse LTR (LTR_{rev}) techniques. In the case of LTR_{forw}, the compact representation of features is computed from the first frame to the last frame in a sequence, whereas in the case of LTR_{rev}, the representation is computed from the last frame to the first frame. With ResNet101,

TABLE 2. Confusion matrix for using ResNet101 features with the LTR method for five frames. The average accuracy is 95.28%.

		Predicted		
		Heavy	Medium	Light
True	Heavy	39	5	0
	Medium	2	41	2
	Light	0	3	162

LTR_{forw} and LTR_{rev} yield accuracy values of 95.28% and 95.10%, respectively. Both forward and backward representations yield comparable accuracy values. The concatenation of LTR_{forw} and LTR_{rev} also yields an accuracy of 95.28%. Additionally, we compared the performance of LTR to two state-of-the-art temporal pooling methods called max-pooling and average-pooling. These pooling methods were also applied to the features extracted from each frame with the goal of generating a compact description for an entire input sequence. We obtained accuracy values of 94.42% and 94.56% with max-pooling and average-pooling, respectively, which are lower than the accuracy achieved with LTR.

Accuracy of motion trajectory features: Corner points are extracted and tracked across video frames to derive point trajectories. The proposed algorithm for removing background points while retaining vehicle points is applied to the trajectories. In each frame, the ROI is divided into 4×4 blocks, and the average velocity of the points is computed in each block. In this experiment, classification decisions were only calculated once, meaning the average velocity was calculated for an entire frame. The resulting accuracy when employing the proposed motion features was 95.67%. The confusion matrix for using motion features is presented in Table 3. One can see that 4 heavy traffic videos are misclassified as medium traffic videos.

TABLE 3. Confusion matrix for using the proposed motion features. The average accuracy is 95.67%.

		Predicted		
		Heavy	Medium	Light
True	Heavy	39	4	1
	Medium	2	41	2
	Light	1	1	163

Accuracy of the proposed method: Table 4 compares the accuracy of the proposed method to those of several

Methods	Samples		
			
	"cctv..620x00106"	"cctv..615x00030"	"cctv..615x00041"
Motion-based classification	Light	Medium	Heavy
CNN-based classification	Light	Medium	Heavy
Proposed method	Light	Medium	Heavy
Ground truth	Light	Medium	Heavy
			
	"cctv..613x00009"	"cctv..516x01641"	"cctv..615x00040"
Motion-based classification	Light	Heavy	Heavy
CNN-based classification	Medium	Medium	Medium
Proposed method	Light	Medium	Heavy
Ground truth	Light	Medium	Heavy
			
	"cctv..516x01639"	"cctv..617x00072"	"cctv..517x01655"
Motion-based classification	Medium	Heavy	Heavy
CNN-based classification	Heavy	Medium	Medium
Proposed method	Heavy	Heavy	Medium
Ground truth	Medium	Medium	Heavy

FIGURE 6. Performance of the proposed method with the UCSD dataset.

TABLE 4. Comparison of the performances of several methods with the UCSD dataset.

Method	Accuracy (%)
Asmaa [16]	95.58
Riaz [3]	95.28
Luo [4]	96.90
Ribas [37]	96.06
Wang [38]	93.30
Texture features	95.28
Motion trajectory features	95.67
Proposed method	**97.64**

state-of-the-art methods. From the table, it is observed that our method yields the best accuracy. The confusion matrix for the proposed method is presented in Table 5. One can see that the numbers of misclassified heavy and light traffic videos drop to two and one, respectively. Only six traffic videos are misclassified in total. These results demonstrate that the proposed hybrid method outperforms texture- and motion-based methods alone.

It is important to note that the UCSD videos were captured during the daytime under various conditions, such as clear, overcast, and rainy weather. Fig. 6 presents the results for samples from the UCSD dataset under various conditions. The classification results for the motion, texture, and hybrid classifiers are presented for each sample and compared to the GT. Examples of total agreement between the

Methods	Samples		
			
	Frame 19205	Frame 1140	Frame 2705
Motion-based classification	Light	Medium	Heavy
CNN-based classification	Light	Medium	Heavy
Proposed method	Light	Medium	Heavy
			
	Frame 19080	Frame 1075	Frame 8740
Motion-based classification	Light	Light	Heavy
CNN-based classification	Medium	Medium	Medium
Proposed method	Light	Medium	Heavy

FIGURE 7. Performance of the proposed method on the NU1 video. The first row presents frame examples that are consistent for both motion and texture classifiers. In the second row, examples of misclassifications for one of the two classifiers are presented. The proposed hybrid classifier re-corrects the final decisions.

TABLE 5. Confusion matrix for the proposed hybrid method with an average accuracy of 97.64%.

		Predicted		
		Heavy	Medium	Light
True	Heavy	42	2	0
	Medium	2	42	1
	Light	0	1	164

three classifiers and the GT are presented in the first row. In the second row, examples of inconsistent decisions between the motion and texture classifiers are presented while the proposed method yields the correct result. In the third row, three video samples (from six videos) are presented as misclassification results.

It is worth noting that we studied the combination of both texture and motion features (i.e., feature-level fusion) and the use of only one SVM classifier. However, the obtained accuracy was less than that achieved by the aggregation of separate SVM classifiers. This is because using separate feature classifiers (texture and motion) allows us to classify videos from different perspectives. For example, a motion-based classifier discriminates between traffic videos from the perspective of vehicle motions, whereas a texture-based classifier distinguishes different traffic videos according to vehicle density. Additionally, we tested the performance of different machine learning classification algorithms in our experiments, including k-nearest neighbors, bagging, and decision tree classifiers. These methods yielded accuracy rates of less than 93% and 95% for texture and motion features, respectively. Therefore, our experimental results demonstrate that the use of separate motion and texture features for training SVM classifiers and the aggregation of their predictions yields enhanced performance.

B. EXPERIMENTS ON THE NU1 VIDEO

In this experiment, we considered the NU1 video to assess the capability of the proposed method to obtain fast and accurate traffic congestion classification results for long videos of poor quality. The NU1 video consists of 40826 frames with a duration of 45 min. The NU1 video was recorded on a highway, where the height of the installed camera was six meters. In this video, 3337 vehicles were counted. In addition to the poor quality of this video, another difficulty is that

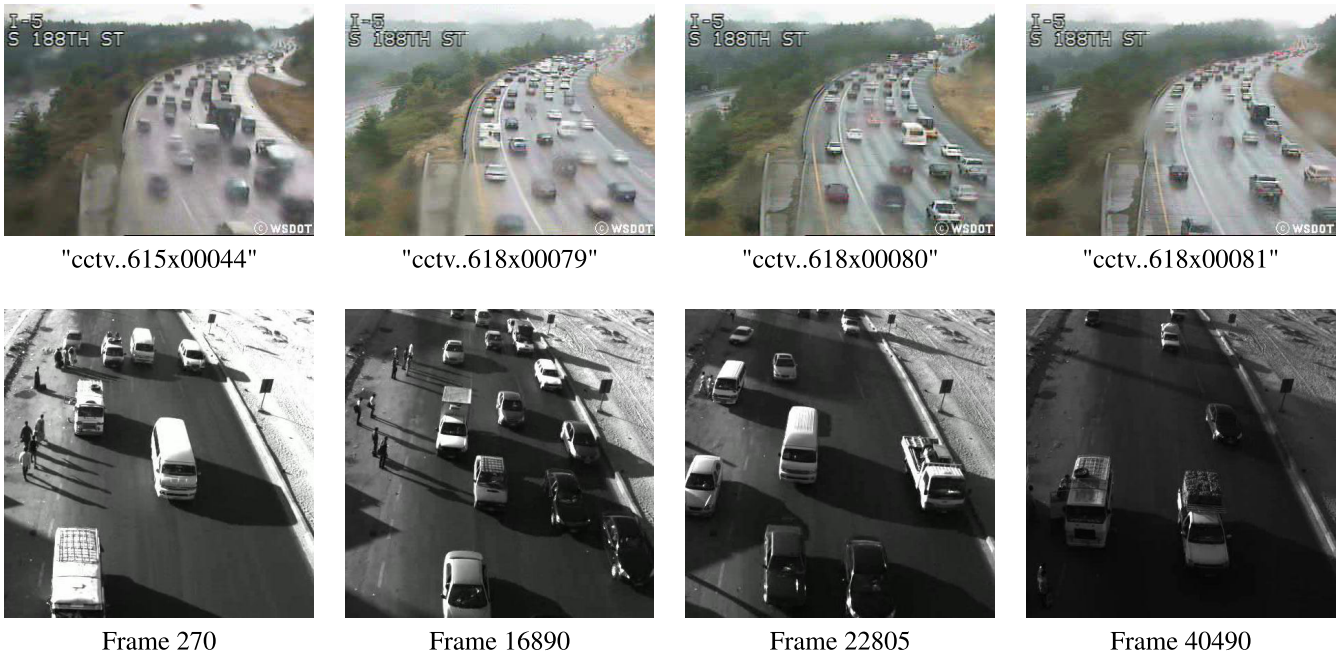

FIGURE 8. Examples of traffic videos with poor illumination. The first row presents video examples from the UCSD dataset in which illumination is affected by rain or overcast weather. In the second row, shadow variation samples from the NU1 video are presented.

vehicles do not follow any lanes or fixed patterns. The traffic congestion in this video varies between low, normal, and heavy.

No new training was performed because we used the SVM classifiers trained using the UCSD dataset in the previous experiments. For every five frames in the NU1 video, motion and texture features were extracted. The ResNet101 network was used for extracting texture vectors, and LTR was used to combine the ResNet101 features from every five frames into compact vectors. Next, the motion and texture features were inputted into the pre-trained SVM classifiers to predict traffic congestion. First, we present qualitative analysis of the classification results. Fig. 7 presents some samples of traffic classification results. Underneath each sample, classifications based on motion trajectories, texture features, and the proposed method are presented. The results of the proposed method are more accurate than those of texture or motion features alone and closely match the actual traffic congestion levels. Although the SVM model was trained on the UCSD dataset, which contains different scenes compared to the NU1 dataset, the obtained results are accurate because the proposed method models video dynamics based on deep residual learning and motion trajectories.

Additionally, we randomly selected ten batches of N frames, $N = 5$ from the NU1 video and had computer vision researchers assign labels to each group describing the statuses of traffic congestion (low, medium, or heavy). We also used the proposed model to predict the statuses of traffic congestion for the same ten batches. We used Cohen's kappa coefficient to measure the agreement between the results of the experts and those of the proposed classifier. We found that $\kappa = 0.849$, indicating strong agreement between the results.

C. ROBUSTNESS OF THE PROPOSED APPROACH TO ILLUMINATION CHANGES

In this section, we focus on analyzing the robustness of the proposed method to illumination changes. Fig. 8 presents examples of poor illumination in the videos considered in our experiments. The video samples in the first row are from the UCSD dataset. The illumination in these videos is heavily influenced by poor weather (rainy or overcast). Despite the low quality of these videos, the proposed approach successfully classifies all these videos. In the second row of Fig. 8, the sample frames come from the NU1 video. In these video frames, the shadows change significantly, leading to large illumination variations between frames. Additionally, occlusion of various parts of vehicles occurs in some frames based on the presence of shadows, as shown in frame 40490 (Fig. 8, row 2, right). The proposed approach can also classify traffic congestion in these video frames accurately.

Table 6 presents analysis of the classification results obtained for the UCSD videos for three weather conditions (rainy, overcast, and clear). In the UCSD dataset, there are 51 videos with clear weather, 125 videos with overcast weather, and 78 videos with rainy weather. The number of misclassified videos and classification accuracy are provided for each weather condition. One can see that the proposed method achieves accuracy rates greater than 96% for all three weather conditions. The proposed method misclassifies two videos with clear weather, three videos with overcast weather, and one video with rainy weather. This analysis indicates

TABLE 6. Analysis of the classification results for UCSD videos according to weather conditions.

Weather	Total number	Correct	Misclassified	Accuracy (%)
Clear	51	49	2	96.08
Overcast	125	122	3	97.60
Rainy	78	77	1	98.72

that both weather conditions and the difficulty of a scene itself (e.g., number of vehicles and occlusion) may affect the classification of traffic videos.

All experiments were conducted using MATLAB 2019a running on a 64 bit Ubuntu operating system with a 3.4 GHz Intel Core-i7 CPU, 16 GB of RAM, and an Nvidia GTX 1070 GPU with 8 GB of video RAM. The average CPU times for extracting CNN features, motion features, LTR, and SVM are 8.1, 1.8, 0.23, and 5.7 ms, respectively. The total CPU time required for the proposed method is less than 16 ms per frame for classifying traffic congestion.

The processing times for the aforementioned steps of the proposed method for all three traffic conditions (heavy, medium, and light) and weather conditions (rainy, overcast, and clear)were checked. We found that the average processing time does not change significantly with traffic or weather conditions for any step. The processing time of the proposed method is lower than those of the SegCNN models reported in [5] (59 ms). Therefore, the proposed method can be used to detect traffic congestion in real time.

The main advantages of the proposed method can be summarized as follows. 1) It provides accurate traffic congestion classification results under different weather conditions (accuracy of 97.64%). 2) Its processing time is very short (less than 16 ms). 3) It can be integrated with expert systems to discover key congestion points for urban traffic and enhance the efficiency of transportation networks. 4) The structure and weights of the proposed model trained on public datasets can be transferred for local re-training (i.e., videos collected from a particular region in the world). After fine tuning the learning process, a re-trained model with an established local dataset can detect traffic congestion with high accuracy.

In this study, the proposed method was tested on two datasets with fixed scenes. The performance of the proposed method can be assessed in future studies using additional traffic congestion datasets with varying scenes.

V. CONCLUSION

In this article, a rapid and reliable traffic congestion classification approach was proposed based on the modeling of video dynamics using a deep residual network and motion trajectories. Unlike existing texture-based traffic congestion classification approaches, the proposed method uses the LTR method to capture the latent structures of traffic videos instead of simply extracting texture features from individual frames. This yields a compact and accurate description for every batch of frames. In our study, we assessed the performance of various deep CNNs and handcrafted texture analysis methods for the feature extraction task. We found that extracting features using the ResNet101 model yields the highest accuracy based on its powerful representational ability. Regarding motion features, we introduced a practical algorithm for discriminating the motion features of stopped vehicles from those of the background; most existing motion-based traffic congestion classification approaches cannot overcome this challenge.

Experimental results demonstrated that the proposed method provides reliable and rapid traffic congestion detection results. We found that obtaining traffic congestion results for batches of five frames yields accurate results with a very short processing time (16 ms), meaning the proposed method is sufficient for traffic analytics. We determined that the use of the proposed motion-based or texture-based methods separately yields classification accuracies greater than 95% for the UCSD dataset, but the proposed hybrid method achieves an accuracy of 97.64%. Additionally, the traffic classification model trained on the UCSD dataset was assessed on a long video (NU1 dataset) with large variations in shadow intensity. The obtained results confirm the reliability and robustness of the proposed method. Overall, the proposed method can achieve accurate traffic congestion classification results under different weather conditions rapidly, meaning it can be used for real-time applications.

Future work will focus on assessing the performance of the proposed method on additional traffic congestion datasets with varying scenes. This comprehensive assessment should help us further improve the performance of the proposed method. In addition, we will employ different aggregation techniques to integrate texture and motion features to improve the accuracy of traffic congestion classification results further.

ACKNOWLEDGMENT

(Mohamed A. Abdelwahab and Mohamed Abdel-Nasser contributed equally to this work.)

REFERENCES

[1] A. M. de Souza, C. A. Brennand, R. S. Yokoyama, E. A. Donato, E. R. Madeira, and L. A. Villas, "Traffic management systems: A classification, review, challenges, and future perspectives," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 4, Apr. 2017, Art. no. 155014771668361.

[2] S. R. E. Datondji, Y. Dupuis, P. Subirats, and P. Vasseur, "A survey of vision-based traffic monitoring of road intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 10, pp. 2681–2698, Oct. 2016.

[3] A. Riaz and S. A. Khan, "Traffic congestion classification using motion vector statistical features," in *Proc. 6th Int. Conf. Mach. Vis. (ICMV)*, vol. 9067, 2013, Art. no. 90671A.

[4] Z. Luo, P.-M. Jodoin, S.-Z. Li, and S.-Z. Su, "Traffic analysis without motion features," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2015, pp. 3290–3294.

[5] Z. Luo, P.-M. Jodoin, S.-Z. Su, S.-Z. Li, and H. Larochelle, "Traffic analytics with low-frame-rate videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 28, no. 4, pp. 878–891, Apr. 2018.

[6] A. B. Chan and N. Vasconcelos, "Classification and retrieval of traffic video using auto-regressive stochastic processes," in *Proc. IEEE Intell. Vehicles Symp.*, 2005, pp. 771–776.

[7] J. B. Kim, C. W. Lee, K. M. Lee, T. S. Yun, and H. J. Kim, "Wavelet-based vehicle tracking for automatic traffic surveillance," in *Proc. IEEE Region 10 Int. Conf. Electr. Electron. Technol. (TENCON)*, vol. 1, Aug. 2001, pp. 313–316.

[8] L. Huang and M. Barth, "Real-time multi-vehicle tracking based on feature detection and color probability model," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2010, pp. 981–986.
[9] G. Mo and S. Zhang, "Vehicles detection in traffic flow," in *Proc. 6th Int. Conf. Natural Comput.*, Aug. 2010, pp. 751–754.
[10] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.
[11] H. Yang and S. Qu, "Real-time vehicle detection and counting in complex traffic scenes using background subtraction model with low-rank decomposition," *IET Intell. Transp. Syst.*, vol. 12, no. 1, pp. 75–85, Feb. 2018.
[12] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.
[13] K. Singh and P. Jain, "Traffic control enhancement with video camera images using AI," in *Optical and Wireless Technologies*. Cham, Switzerland: Springer, 2020, pp. 137–145.
[14] M. A. Abdelwahab, "Fast approach for efficient vehicle counting," *Electron. Lett.*, vol. 55, no. 1, pp. 20–22, Jan. 2019.
[15] F. Porikli and X. Li, "Traffic congestion estimation using HMM models without vehicle tracking," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2004, pp. 188–193.
[16] O. Asmaa, K. Mokhtar, and O. Abdelaziz, "Road traffic density estimation using microscopic and macroscopic parameters," *Image Vis. Comput.*, vol. 31, no. 11, pp. 887–894, Nov. 2013.
[17] M. A. Abdelwahab, "Accurate vehicle counting approach based on deep neural networks," in *Proc. Int. Conf. Innov. Trends Comput. Eng. (ITCE)*, Feb. 2019, pp. 1–5.
[18] A. Gomaa, M. M. Abdelwahab, M. Abo-Zahhad, T. Minematsu, and R.-I. Taniguchi, "Robust vehicle detection and counting algorithm employing a convolution neural network and optical flow," *Sensors*, vol. 19, no. 20, p. 4588, Oct. 2019.
[19] P. Wang, W. Hao, Z. Sun, S. Wang, E. Tan, L. Li, and Y. Jin, "Regional detection of traffic congestion using in a large-scale surveillance system via deep residual trafficnet," *IEEE Access*, vol. 6, pp. 68910–68919, 2018.
[20] Z. Sun, P. Wang, J. Wang, X. Peng, and Y. Jin, "Exploiting deeply supervised inception networks for automatically detecting traffic congestion on freeway in China using ultra-low frame rate videos," *IEEE Access*, vol. 8, pp. 21226–21235, 2020.
[21] S. Zhang, S. Li, X. Li, and Y. Yao, "Representation of traffic congestion data for urban road traffic networks based on pooling operations," *Algorithms*, vol. 13, no. 4, p. 84, Apr. 2020.
[22] B. Fernando, E. Gavves, D. Muselet, and T. Tuytelaars, "Learning to rank based on subsequences," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 2785–2793.
[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.
[25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.
[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[27] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 886–893.
[28] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.
[29] A. Ramirez Rivera, J. Rojas Castillo, and O. Oksam Chae, "Local directional number pattern for face analysis: Face and expression recognition," *IEEE Trans. Image Process.*, vol. 22, no. 5, pp. 1740–1752, May 2013.
[30] B. Fernando, E. Gavves, J. Oramas M., A. Ghodrati, and T. Tuytelaars, "Rank pooling for action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 773–787, Apr. 2017.
[31] A. Saleh, M. Abdel-Nasser, M. Angel Garcia, and D. Puig, "Aggregating the temporal coherent descriptors in videos using multiple learning kernel for action recognition," *Pattern Recognit. Lett.*, vol. 105, pp. 4–12, Apr. 2018.
[32] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 1994, pp. 593–600.
[33] M. A. Abdelwahab and M. M. Abdelwahab, "A novel algorithm for vehicle detection and tracking in airborne videos," in *Proc. IEEE Int. Symp. Multimedia (ISM)*, Dec. 2015, pp. 65–68.
[34] S. A. Aly, A. Mamdouh, and M. Abdelwahab, "Vehicles detection and tracking in videos for very crowded scenes," in *Proc. MVA*, 2013, pp. 311–314.
[35] D. Impedovo, F. Balducci, V. Dentamaro, and G. Pirlo, "Vehicular traffic congestion classification by visual features and deep learning approaches: A comparison," *Sensors*, vol. 19, no. 23, p. 5213, Nov. 2019.
[36] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proc. 5th Annu. Workshop Comput. Learn. Theory (COLT)*, 1992, pp. 144–152.
[37] L. C. Ribas, W. N. Goncalves, and O. M. Bruno, "Dynamic texture analysis with diffusion in networks," 2018, *arXiv:1806.10681*. [Online]. Available: http://arxiv.org/abs/1806.10681
[38] Y. Wang, L. Wang, D. Kong, and B. Yin, "Extrinsic least squares regression with closed-form solution on product Grassmann manifold for video-based recognition," *Math. Problems Eng.*, vol. 2018, pp. 1–7, Mar. 2018.

MOHAMED A. ABDELWAHAB received the Ph.D. degree in electronics and communication engineering from the Egypt Japan University of Science and Technology (E-JUST), Egypt, in 2016. He is currently a Postdoctoral Researcher with the Center of Coevolutionary Research for Sustainable Communities, Kyushu University, Japan. Since 2016, he has been an Assistant Professor with the Department of Electrical Engineering, Faculty of Energy Engineering, Aswan University. He is the author of more than 16 articles. His research interests include image processing, computer vision, vehicle detection and tracking, traffic monitoring, and pedestrian behavior analysis.

MOHAMED ABDEL-NASSER received the Ph.D. degree in computer engineering from Universitat Rovira i Virgili, Spain, in 2016. He was the Manager of the E-learning and Digital Library Centre, Aswan University, Egypt, in 2018. He is currently a Postdoctoral Researcher with Universitat Rovira i Virgili and an Assistant Professor with the Department of Electrical Engineering, Aswan University. He has participated in several projects funded by the European Union, the Government of Spain, and the Government of Egypt. He has published more than 65 papers in international journals and conferences. His research interests include the application of machine learning and deep learning to several real-world problems, including smart road environment, medical image analysis, smart grid analysis, and time-series forecasting. He received the Marc Esteva Vivanco Prize for the Best Ph.D. Dissertation on Artificial Intelligence. He is also a Co-Founder of the International Conference on Innovative Trends in Computer Engineering (ITCE) and the Co-Chair of the Information Technology Track. He is also an active reviewer for many prestigious journals, such as IEEE TRANSACTIONS ON MEDICAL IMAGING, IEEE JOURNAL OF BIOMEDICAL AND HEALTH INFORMATICS, IEEE ACCESS, *Expert Systems with Applications*, and *Neural Computing and Applications*.

MAIYA HORI (Member, IEEE) received the Ph.D. degree in information science from the Nara Institute of Science and Technology, in 2011. He is currently an Associate Professor with the Platform of Inter/Transdisciplinary Energy Research, Kyushu University. His research interests include computer vision and machine learning.