Hindawi
Discrete Dynamics in Nature and Society
Volume 2017, Article ID 8523495, 9 pages
https://doi.org/10.1155/2017/8523495

Research Article

Learning to Detect Traffic Incidents from Data Based on Tree Augmented Naive Bayesian Classifiers

Dawei Li,[1] Xiaojian Hu,[1] Cheng-jie Jin,[1] and Jun Zhou[2]

[1]*Jiangsu Key Laboratory of Urban ITS, School of Transportation, Southeast University, Jiangsu Province Collaborative Innovation Center of Modern Urban Traffic, Sipailou No. 2, Xuanwu District, Nanjing 210096, China*
[2]*Huaiyin Institute of Technology, Key Laboratory for Traffic and Transportation Security of Jiangsu Province, Meicheng Rd, Huaian 223003, China*

Correspondence should be addressed to Dawei Li; lidawei@seu.edu.cn

Received 25 April 2017; Revised 17 July 2017; Accepted 27 July 2017; Published 2 October 2017

Academic Editor: Gabriella Bretti

This study develops a tree augmented naive Bayesian (TAN) classifier based incident detection algorithm. Compared with the Bayesian networks based detection algorithms developed in the previous studies, this algorithm has less dependency on experts' knowledge. The structure of TAN classifier for incident detection is learned from data. The discretization of continuous attributes is processed using an entropy-based method automatically. A simulation dataset on the section of the Ayer Rajah Expressway (AYE) in Singapore is used to demonstrate the development of proposed algorithm, including wavelet denoising, normalization, entropy-based discretization, and structure learning. The performance of TAN based algorithm is evaluated compared with the previous developed Bayesian network (BN) based and multilayer feed forward (MLF) neural networks based algorithms with the same AYE data. The experiment results show that the TAN based algorithms perform better than the BN classifiers and have a similar performance to the MLF based algorithm. However, TAN based algorithm would have wider vista of applications because the theory of TAN classifiers is much less complicated than MLF. It should be found from the experiment that the TAN classifier based algorithm has a significant superiority over the speed of model training and calibration compared with MLF.

1. Introduction

Traffic congestion is a challenging problem in most of the big cities all over the world. Congestion leads to increasing traffic delays, higher fuel consumption, and negative environmental effects. The cost of total delay caused by traffic congestion in rural and urban areas is estimated to be around $1 trillion per year in the United States [1]. Traffic congestion can be classified into two categories: recurrent congestion generated by excess demand and unrecurrent congestion caused by incidents. Some studies have estimated that around 60% of all traffic congestion on highways is caused by incidents [2]. Due to the large losses caused by unrecurrent congestion, incident manage system (IMS) has been a key component of the advanced traffic management system now.

As a core technique in IMS, automatic incident detection (AID) algorithms have also become an interesting and active research topic. A considerable amount of research has addressed this problem and several techniques have been developed over the last few decades. Depending on the methodology, an algorithm is usually classified into one of five major categories: comparative algorithms, statistical algorithms, time-series and filtering based algorithms, traffic theory based algorithms, and advanced algorithms.

The California algorithms are classic examples of the comparative algorithms, which are one of the first widely implemented incident detection algorithms developed [3]. They are usually used as benchmarks for evaluating the performance of other algorithms. Examples of the statistical algorithms include the standard normal deviate (SND) algorithm [4] and the Bayesian Algorithms [5], which use standard statistical techniques to identify sudden changes in the traffic flow parameters. Time-series and filtering algorithms treat the traffic flow parameters as time-series. The

deviation from the modeled time-series behavior is used for indication of incidents. The moving average (MA) algorithm [6] and the exponential smoothing based algorithms [7] are included in this category. One classic example of traffic theory based algorithms is McMaster algorithm, which is based on the catastrophe theory. McMaster algorithm determines the state of traffic based on its position in the flow-density plot and detects incidents based on the transition of the point from one state to another [8]. In order to improve the detection performance, the latest trend has been the development of advanced algorithms, which are based on advanced mathematical formulation. Neural Networks based algorithms [9, 10], which have an attractive performance in the lab environment, are classified into this category.

By reviewing the existing incident detection algorithms, it should be noted that most of these algorithms, which perform well in the lab environment, have not been used in practice. The biggest reason is that, it is usually difficult to transfer these algorithms from site to site and keep a well performance. To meet the universality requirements for which the advanced traffic management systems called, the Bayesian networks have been used to develop universal algorithms [11, 12]. According to the testing results, these algorithms demonstrate very stable performance and strong transferability. However, in the previous studies, the Bayesian networks based algorithms strongly depend on the knowledge of experts. For instance, the determination of the Bayesian network structure and the discretization of continuous attributes are both artificially predetermined in these studies.

As a special case of Bayesian network, the naive Bayesian classifier does not need the predetermination of network structure [13]. However, the naive Bayesian classifier has too strict assumptions on the independent relations of variables. To improve the performances, many improved naive Bayes algorithms are proposed in literature, such as the Averaged One-Dependence Estimators (AODE) [14], Weighted Average of One-Dependence Estimators (WAODE) [15], Hidden Naive Bayes (HNB) [16], Deep Feature Weighted Naive Bayes (DFWNB) [17], Discriminatively Weighted Naive Bayes (DWNB) [18], Averaged Tree Augmented Naive Bayes (ATAN) [19], and Super Parent TAN (SP) [14].

In this study, in order to reduce the dependency on experts' knowledge, a tree augmented naive Bayesian (TAN) classifier, which is a special form of Bayesian networks, is chosen to develop an incident detection algorithm in this paper. The TAN classifier can reduce the dependency on experts' knowledge, because the structure and parameters of the proposed TAN classifier are both learned from the data, and an entropy-based method is proposed for the discretization of continuous variables completely depending on the data. The performance of this algorithm would also be evaluated using a simulation dataset.

The remaining part of the article is structured as follows: Section 2 is a simple introduction of TAN classifier. Section 3 presents the procedure of data preprocessing. Section 4 discusses the development and implementation of the TAN classifier for incident detection. In Section 5, an experiment of this algorithm with a simulation dataset is carried out. Finally, Section 6 gives some conclusions and remarks.

2. TAN Classifier

2.1. Bayesian Networks. Since TAN classifier is a special case of Bayesian networks, it is necessary to introduce Bayesian networks first. Bayesian networks are directed acyclic graphs that allow efficient and effective representation of the joint probability distribution over a set of random variables [20]. Formally, a Bayesian network for a set of random variables $\mathbf{U} = \{X_1, \ldots, X_n\}$ is a pair $B = (G, \boldsymbol{\theta})$. The first component, G, is a directed acyclic graph whose vertices correspond to the random variables $X_1, \ldots, X_n$, and whose edges represent direct dependencies between the variables. The graph G, which is also called the structure of this Bayesian network, encodes independence assumption: each variable X_i is independent of its no-descendants given its parents in G. The second component of the pair represents the set of parameters that quantifies the network. It contains a parameter $\theta_{X_i|\Pi_{x_i}} = P_B(x_i \mid \Pi_{X_i})$ for each possible value x_i of X_i, and Π_{x_i} of Π_{X_i}, where Π_{X_i} denotes the set of parents of X_i in G. A Bayesian network B defines a unique joint probability distribution over $\mathbf{U}$ given by

$$P_B(X_1, \ldots, X_n) = \prod_{i=1}^{n} P_B\left(X_i \Pi_{X_i}\right) = \prod_{i=1}^{n} \theta_{X_i|\Pi_{X_i}}. \quad (1)$$

If a Bayesian network is used for incident detection, it must contain a class variable INC and attribute variables $A_1, \ldots, A_n$. INC = 1 means an incident has happened. $A_1, \ldots, A_n$ denote traffic flow parameters. When it is implemented in IMS, $A_1, \ldots, A_6$ are input in real time, and this Bayesian network is used to update the post probability $P(\text{INC} = 1 \mid A_1, \ldots, A_n)$.

Because of the independence assumption, in a Bayesian network for incident detection, the class variable INC must get links to all the attribute variables; therefore each and every piece of information about attribute variables can be made use of to update the incident probability. Therefore, the structure of this Bayesian network cannot be learned using any unrestricted learning algorithms. In the previous studies, it is manually determined according to experts knowledge. Figure 1(a) presents a possible structure of the Bayesian networks for incident detection.

2.2. Naive Bayesian (NB) Classifier. As analyzed above, in a Bayesian network for incident detection, the class variable INC must get links to all attribute variables. Therefore, in order to determine the structure of this Bayesian network, the only remaining problem is how to add the edges between the attribute variables. The simplest way to solve this problem is to assume that every attribute $(A_1, \ldots, A_n)$ is independent from the rest of the attributes, given the value of the class variable. Thus, there is no edges between each pair of attribute variables. This form of Bayesian networks is often referred to as naive Bayesian (NB) classifier. The structure of NB classifier for incident detection can be demonstrated in Figure 1(b).

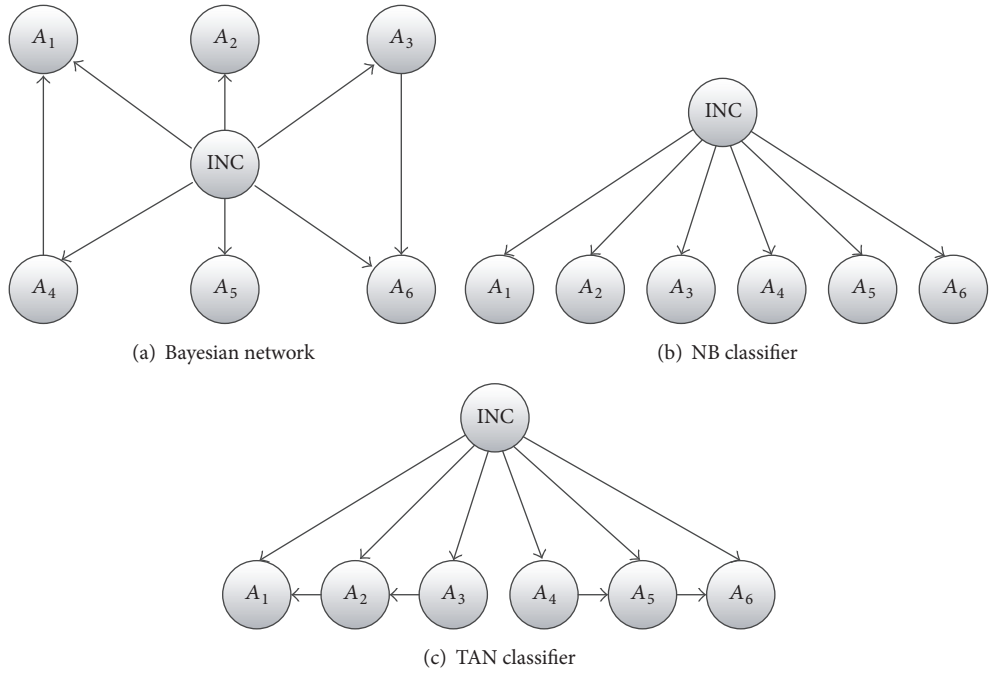

Figure 1: Typical structures of Bayesian network, NB classifier, and TAN classifier for incident detection.

2.3. TAN Classifier. Although the performance of NB classifiers is surprising in some practical cases, the main assumption behind them is clearly unrealistic. Accordingly, a class of network structures based on the NB classifier, which are called tree augmented naive Bayesian (TAN) classifiers, is selected to develop incident detection algorithm in this paper. In a TAN classier, as shown in Figure 1(c), the class variable INC has no parent and each attribute has the class variable and at most one other attribute as parents. This ensures that, in the learned network, the probability $P(\text{INC} \mid A_1, \ldots, A_n)$ will take all attributes into account, and the strong assumption of independence in NB classifier is relaxed. The edges between attribute variables can be learned from the training dataset.

3. Data Preprocessing

The raw traffic measurements collected by the detectors cannot be used to develop and implement detection algorithms without a procedure of data preprocessing. This procedure contains three parts in this study: wavelet denoising, normalization, and discretization.

3.1. Wavelet Denoising. The purpose of wavelet denoising is to eliminate the random fluctuation from raw traffic measurements. The random fluctuation is caused by the randomness of drivers' behaviors and the measurement errors of detectors. In most of the previous studies, the raw traffic measurements are smoothed by moving average method. Although this method is very simple in theory and easy to be implemented, it will increase the mean time to detection (MTTD) because the sudden changes in the traffic flow parameters caused by incidents are also smoothed. Accordingly, the wavelet denoising is chosen instead of smoothing method in this study.

The general wavelet denoising procedure is as follows.

Step 1. Select appropriate wavelet function and apply wavelet transform to the noisy signal to produce the noisy wavelet coefficients to the level which can meet the requirements.

Step 2. Select appropriate threshold limit at each level and threshold method (hard or soft thresholding) to best remove the noises.

Step 3. Inverse wavelet transform of the thresholded wavelet coefficients to obtain a denoised signal.

In this procedure, there are three things that should be decided first for the wavelet denoising process: wavelet function, level of decomposition, and threshold method.

When applied in the detection algorithm, the raw traffic measurements obtained from the previous $N - 1$ time intervals to the present time interval t, a sequence contains N elements ($N = 16$ in this study), is wavelet denoised. In the next time interval $t + 1$, the raw traffic measurement obtained at $t - (N - 1)$ time interval is removed from the sequence, and the measurement obtained at interval $t + 1$ is added to the sequence. A new sequence which still contains N elements is made up, and then it is wavelet denoised again.

3.2. Normalization. After the wavelet denoising process, the denoised traffic measurements should be normalized, which means converting all the measurements to a value range of $[0, 1]$. The normalization process is mainly to enhance the

transferability of algorithm. It should be noted that the same traffic measurements collected in different sites are usually in different value ranges. For instance, the average traffic speeds on a freeway will be in a range of [0, 100] km/h, while the speeds on an arterial road cannot be more than 60 km/h because of the speed limitation. Therefore, if the denoised traffic measurements are used to develop the detection algorithm directly, this algorithm cannot be applied in different roads.

Additionally, the different traffic measurements collected by detectors in the same site are also in different ranges. For instance, the occupancies are in the range of [0, 1], while the maximum value of speeds collected by detectors may be more than 100. Their various orders of magnitude would also be inconvenient to the calculation work.

The normalization process is conducted by

$$a' = \begin{cases} 0, & \text{if } a \le A_\alpha, \\ \dfrac{a - A_\alpha}{A_\beta - A_\alpha}, & \text{if } A_\alpha < a < A_\beta, \\ 1, & \text{if } a \ge A_\beta, \end{cases} \tag{2}$$

where a' is the normalized traffic measurement and a is the denoised traffic measurement. A_α and A_β are the minimum and maximum values of the denoised traffic measurements, respectively, or manually decided according to the frequency histograms of a in the training dataset.

3.3. Discretization. All of the attribute variables in the TAN classifier should be discrete, because the value ranges of the traffic parameters are all continuous. Therefore, the normalized traffic parameters should be compared against the predefined split points to ascertain their states (e.g., volume is high, medium, or low); this process is called discretization.

Instead of depending on the experiences of experts, an entropy-based method is proposed in this paper. Entropy is one of the most commonly used discretization measures. Entropy-based discretization is a supervised, top-down splitting technique. It explores class distribution information in its calculation and determination of split points (data values for partitioning an attribute range). To discretize a normalized traffic measure, A, the method selects the value that minimizes the entropy of the training dataset as a split point and recursively partitions the resulting intervals to arrive at a hierarchical discretization. Let D consist of data tuples defined by a set of attributes and a class-label attribute. The class-label attribute provides the class information per tuple. The basic method for entropy-based discretization of an attribute X within the set is as follows [21].

Step 1. Since the traffic measures are normalized, the value range of A is [0, 1]. From 0 to 1, values are taken at intervals of 0.01, as potential split points (denoted *split_point*) to partition the range of A. That is, a split point for A can partition the tuples in D into two subsets satisfying the conditions $A \le split\_point$ and $A > split\_point$, respectively, thereby creating a binary discretization. The set of all potential split points is denoted by S.

Step 2. Entropy-based discretization, as mentioned above, uses information regarding the class label of tuples. In this case, the labels are the states of traffic; therefore the tuples could be divided into two classes: incident state class (denoted by class C_1) and incident-free class (denoted by class C_2). Suppose we want to classify the tuples in D by partitioning on attribute X and some split point. Ideally, we would like this partitioning to result in an exact classification of the tuples. Which means that we would hope that all of the tuples of class C_1 will fall into one partition, and all of the tuples of class C_2 will fall into the other partition. However, this is unlikely. For example, the first partition may contain many tuples of C_1, but also some of C_2. How much more information would we still need for a perfect classification, after this partitioning? This amount is called the *expected information requirement* for classifying a tuple in D based on partitioning by A. It is given by

$$\text{Info}_A(D) = \frac{|D_1|}{|D|}\text{Entropy}(D_1) + \frac{|D_2|}{|D|}\text{Entropy}(D_2), \tag{3}$$

where D_1 and D_2 correspond to the tuples in D satisfying the conditions $A \le split\_point$ and $A > split\_point$, respectively. $|D|$ is the number of tuples in D, and so on. The entropy function for a given set is calculated based on the class distribution of the tuples in the set. For incident detection, only given 2 classes, C_1 and C_2, the entropy of D_1 is

$$\text{Entropy}(D_1) = -\sum_{i=1}^{2} p_1 \log_2(p_1), \tag{4}$$

where p_i is the probability of class C_i in D_1, determined by diving the number of tuples of class C_i in D_1 by $|D_1|$, the total number of tuples in D_1. The entropy of D_2 can be computed in the same way. Therefore, when selecting a split point for attribute A, it is to find the split point in the set S that minimizes $\text{Info}_A(D)$. Using this split point, the range of A would be partitioned into two intervals, corresponding to $A \le split\_point$ and $A > split\_point$.

Step 3. The process of determining a split point is recursively applied to each partition obtained (to the partition with a lager entropy first), until the number of intervals is greater than a threshold, *max interval*.

After the wavelet denoising, normalization, and discretization processes, the raw traffic measurements are converted to traffic cases that can be input to the TAN classifier.

4. TAN Classifier Based Detection Algorithm

4.1. Learning and Inference. According to the introductions in Section 2, the structure of TAN classifier for incident detection is not given exactly, and the structure with optimal parameter setting and maximal likelihood should be learned from data. Let D be a dataset over the variables $\{A_1, \ldots, A_n, \text{INC}\}$. Using a slight modification of the Chow–Liu algorithm [22], a TAN of maximal likelihood can be constructed as follows.

Step 1. Calculate the conditional mutual information MI $(A_i, A_j \mid \text{INC})$ for each pair (A_i, A_j), where

$$\text{MI}\left(A_i, A_j \mid \text{INC}\right) = \sum_{A_i, A_j} P\left(a_i, a_j, \text{inc}\right) \log \frac{P\left(a_i, a_j \mid \text{inc}\right)}{P\left(a_i \mid \text{inc}\right) P\left(a_j \mid \text{inc}\right)}. \quad (5)$$

Step 2. Consider the complete MI-weighted graph: the complete undirected graph over $\{A_1, \ldots, A_n\}$, where the links $A_i - A_j$ have the weight $\text{MI}(A_i, A_j \mid \text{INC})$.

Step 3. Build a maximal-weight spanning tree for the complete MI-weighted graph.

Step 4. Direct the resulting tree by choosing any variable as a root and setting the directions of the links to be outward from it.

Step 5. Add the node INC and a directed link from INC to each attribute node.

Step 6. Learn the parameters.

In Step 6, the parameter learning algorithm used in this paper is MLE algorithm, which is to choose a parameter estimate $\widehat{\boldsymbol{\theta}}$ that maximizes the likelihood:

$$l(\boldsymbol{\theta} \mid D) = \log \prod_{i=1}^{m} P\left(d_i \mid \boldsymbol{\theta}\right) = \sum_{i}^{m} \log P\left(d_i \mid \boldsymbol{\theta}\right), \quad (6)$$

where $D = \{d_1, d_2, \ldots, d_m\}$ is the training dataset and d_i is the instances of a variable contained in the TAN classifier, respectively.

The purpose of inference is to compute the post probability $P(\text{INC} = 1 \mid A_1, \ldots, A_n)$. The junction tree algorithm is used for inference in this paper. Once the TAN classifier is transformed into a junction tree, the inference involves the following steps [23].

Step 1. Each item of evidence must be incorporated into the junction tree potentials. For each item of evidence, an evidence function is multiplied onto an appropriate clique potential.

Step 2. Some clique is selected. This clique is referred to as the root of the propagation.

Step 3. Then messages are passed toward the selected root. The messages are passed through the separators of the junction tree (i.e., along the links of the tree). These messages cause the potentials of the receiving cliques and separators to be updated. This phase is known as *CollectInformation*.

Step 4. Now messages are passed in the opposite direction (i.e., from the root toward the leaves of the junction tree). This phase is known as *DistributeInformation*.

Step 5. When the calls are completed, the table that contains updated probability distribution of each node is normalized so that it sums to one.

This paper will not describe more details of learning and inference algorithms. The Kevin Murphy's Bayes Net Toolbox (BNT) for MATLAB is used in this research to develop the application programs for incident detection.

4.2. Incident Report. When the TAN classifier is implemented in IMS, $A_1, \ldots, A_6$ are input in real time. After the process of inference, the post probability $P(\text{INC} = 1 \mid A_1, \ldots, A_n)$ can be obtained. If this post probability is compared with a predefined threshold directly, a high false alarm rate (FAR) will be caused. Therefore, in this study, a smoothing method is used to reduce the FAR.

The post probability updated at time interval t is denoted by $I(t)$; the final estimate of incident probability for incident report $\widehat{I}(t)$ can be calculated by

$$\widehat{I}(t) = \alpha I(t) + (1 - \alpha)\, \widehat{I}(t-1), \quad (7)$$

where $\alpha \in [0, 1]$ is a coefficient that can adjust the performance of this detection algorithm. The smaller α will cause lower FAR and higher MTTD, and vice versa. After each time interval, the $\widehat{I}(t)$ is updated and compared with a predefined threshold. If $\widehat{I}(t)$ exceeds the decision threshold, an incident is alarmed. A larger threshold will cause lower FAR and higher detection rate (DR), and vice versa.

5. Experimental Study

5.1. Data Description and Variable Selection. The dataset used in this study is the AYE dataset. This dataset was produced from a traffic simulated system, and it has been used in some researches related to incident detection [10, 24]. A 5.8 km section of the Ayer Rajah Expressway (AYE) in Singapore was selected to simulate incident and incident-free conditions. The simulation system generated volume, occupancy, and speed data upstream and downstream. The traffic dataset consisted of 300 incident cases that had been simulated based on AYE traffic. The simulation of each incident case consisted of three parts. The first part was the incident-free period that lasted for 5 min. This was after a simulation of 5 min warm-up time. The second part was the 10-min incident period. This was followed by a 30 min postincident period. The above 300 incidents were split into two partitions, training data set and testing data set. Each data set had 3000 input patterns for incident state and 10500 patterns corresponding to incident-free state. Each input pattern included traffic volume and speed and lane occupancy accumulated at 30-s intervals, averaged across all the lanes, as well as the label of incident state. In this research, the AYE data set is used to illustrate the procedure of data preprocessing, model development, and performance evaluation of the TAN based detection algorithm.

To develop the TAN classifier for incident detection, the variables contained in the models should be decided at first. According to the contents of datasets and the previous research, eight variables are chosen as the nodes of the TAN classifier; the details are shown in Abbreviations.

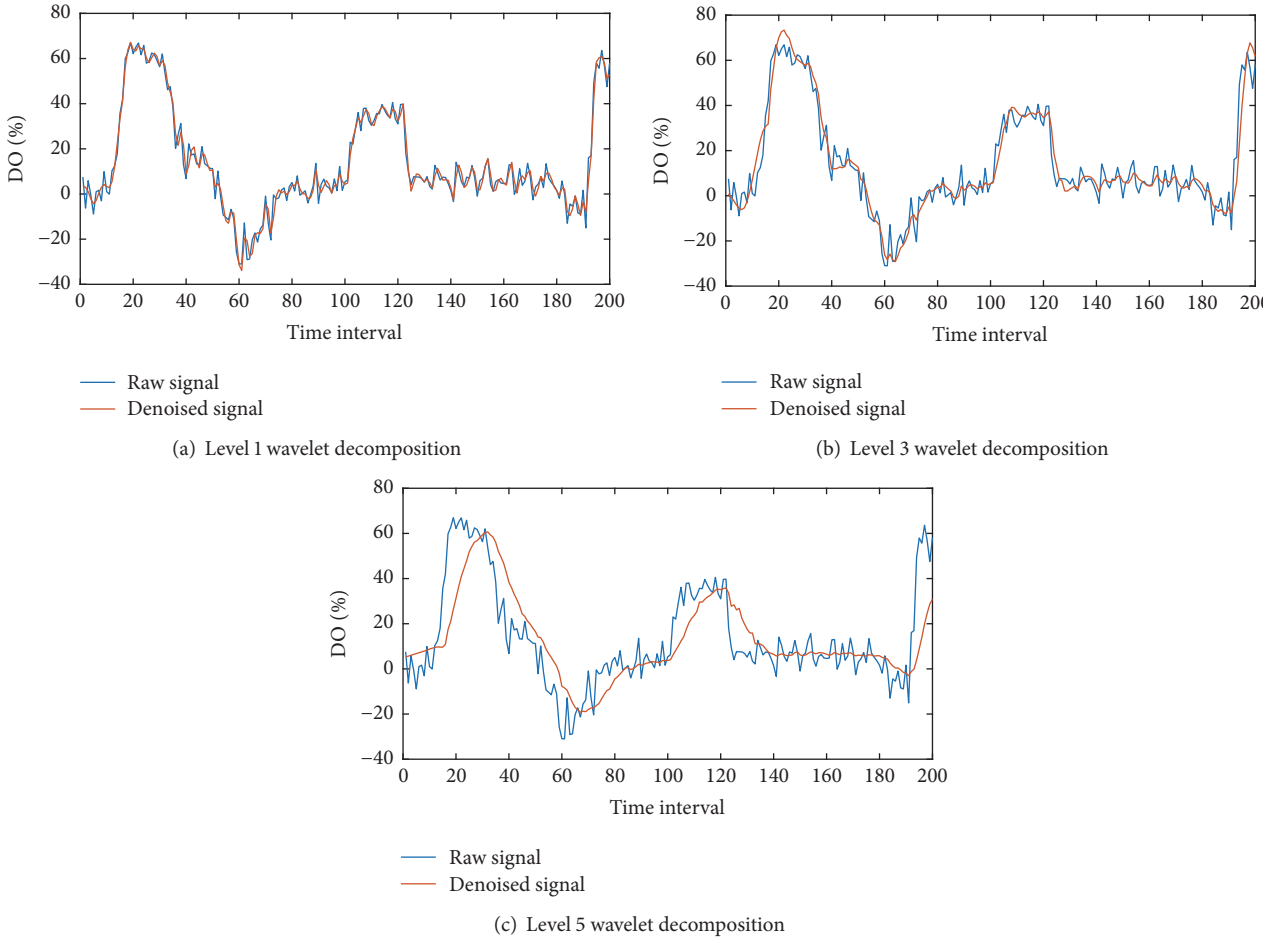

Figure 2: Wavelet denoising results at different levels of decomposition.

5.2. Data Preprocess

5.2.1. Wavelet Denoising. According to the description in Section 3.1, there are three points that should be predetermined for the wavelet denoising process: wavelet function, level of decomposition, and threshold method. In this study, using the wavelet toolbox of MATLAB, *Sym8* wavelet function and *Sqtwolog* soft threshold method are chosen manually to denoise the raw traffic measurements. The remaining question is to decide the level of decomposition. The raw traffic measurements are wavelet denoised with the decomposition level assigned from 1 to 5, and the partial results are shown in Figure 2.

It could be found that the higher level of decomposition will have better denoising performance, but will also increase the computation time and distortions level. According to the experiment results, the optimal decomposition level is 3 in this study. Both the training and testing datasets are wavelet denoised according to the procedure described in Section 3.1.

5.2.2. Normalization. The next step of data preprocess is normalization. According to the frequency histograms of the denoised traffic measures in training dataset, A_α and A_β in (2) could be decided manually and are shown in Table 1. Then both the training and testing datasets are normalized using (2).

Table 1: The values of A_α and A_β in (2) according to training dataset.

	Q_u	Q_d	O_u	O_d	V_u	V_d	DO
A_α	0	0	0	0	0	0	0
A_β	3000	3000	80	50	60	60	80

5.2.3. Discretization. Since increasing the number of variable states will lead to exponential growth of the parameter size of a TAN classifier, all of the traffic flow parameters would be divided into three states. The split points are decided and shown in Table 2 using the training dataset according to the entropy-based discretization procedure described in Section 3.3. Then both the training dataset and testing dataset are discretized with comparing the traffic parameters with the split points.

5.3. Structure Learning.

Using BNT, the structure of TAN classifier for incident detection can be learned from the preprocessed training dataset and shown in Figure 3. As an

Table 2: The split points of continuous variables in TAN classifier.

	Q_u	Q_d	O_u	O_d	V_u	V_d	DO
Split point 1	0.3	0.32	0.62	0.15	0.17	0.6	0.3
Split point 2	0.59	0.5	0.81	0.23	0.3	0.8	0.66

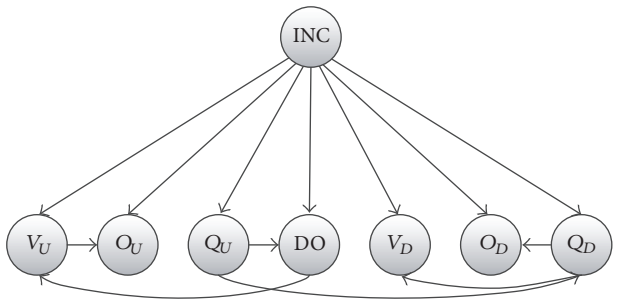

Figure 3: The structure of the TAN classifier for incident detection learned from the training dataset.

example the parameters of node Q_u are shown in Table 3 as an example.

5.4. Evaluation and Analysis

5.4.1. Performance Measures. The common performance measures used to evaluate the incident detection algorithms include the detection rate (DR), the false alarm rate (FAR), and the mean time to detection (MTTD). DR and MTTD are written as

$$\begin{aligned} \text{DR} &= \frac{\text{number of incident cases detected}}{\text{total number of incident cases}}, \\ \text{MTTD} &= \frac{t_1 + \cdots + t_m}{m}, \end{aligned} \tag{8}$$

where t_i is the length of time between the start of the incident and the time the alarm is initiated and m is the number of incident cases detected successfully. If a single incident instance is identified within the period of actual occurrence of one incident case, which often consists of many continuous incident instances, the incident case is regarded as detected.

There are several different formulas for FAR; following is the one used in this study:

$$\text{FAR} = \frac{\text{number of false alarm cases}}{\text{total number of in put instances}}. \tag{9}$$

The number of false alarm cases is computed through taking one instance misclassified as incident instance as one false alarm case.

Both the DR and FAR measure the effectiveness of an algorithm, and the MTTD reflects the efficiency of the algorithm.

5.4.2. Experiment Result with AYE Data. The preprocessed instances in testing dataset are input to the TAN classifier developed in Section 4. The real time incident probability

Table 3: The parameters of Q_u.

Q_u	INC	DO		
		1	2	3
1	1	0.0196	0.3824	0.5980
2	1	0.9243	0.0716	0.0041
3	1	0.9799	0.0201	0.0000
1	2	0.0000	0.1741	0.8259
2	2	0.5861	0.3684	0.0456
3	2	0.9045	0.0955	0.0000

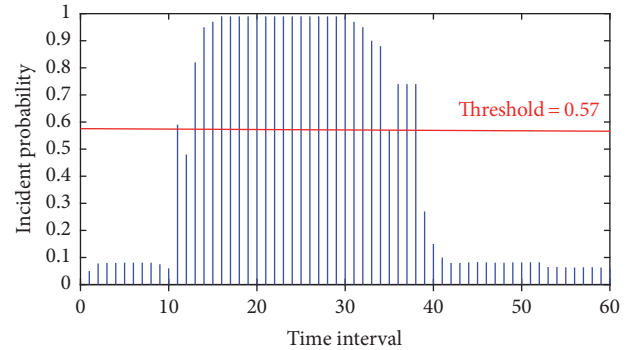

Figure 4: A typical output of TAN classifier based incident detection algorithm.

for incident report $\widehat{I}(t)$ can be calculated by (7) using the output of the TAN classifier. The value of α is assigned as 0.7 in this study. Figure 4 shows a typical output of the incident detection algorithm.

The performance measures are calculated according to the outputs of the algorithm. The TAN based algorithms are tested with different values of the incident report threshold. The testing results are shown in Table 3 compared with neural networks based algorithm and the Bayesian network (BN) based method in previous study [11], which are trained and tested using the same data set.

The neural network models mainly focused on the applications of multilayer feed forward (MLF) neural networks for incident detection. Since the MLF was excellent in the previous researches, it is used as a benchmark for comparison in this paper. The MLF based algorithms used in this paper are developed and evaluated by Wang et al. [24] using the same AYE data, and this paper just refers to their work. In their research, 10 network classifiers are trained using the training data set containing 50.00% of incident instances, and they tested them on the testing data set. The networks fall into two different structures, one has three layers with six neurons in input layer, three neurons in hidden layer, and one neuron in output layer; another has the same number of input neurons and output neuron but has six neurons in hidden layer. The output indicates the traffic state compared to a predefined threshold, here set as zero; that is, if it is larger than 0, it indicates the occurrence of incident for this instance otherwise nonincident. The parameters for training network were set as follows: learning rate is 0.1, the maximal

Table 4: Comparison between TAN and MLF with AYE data.

Algorithms	Thresholds	DR (%)	FAR (%)	MTTD (seconds)
TAN	0.7	90.73	1.90	135
	0.6	95.97	3.86	129
	0.5	95.71	4.54	75
BN	0.7	89.25	1.85	90
	0.6	92.34	2.98	84
	0.5	92.29	3.04	67
MLF	—	98.80	5.88	84

training epochs are 1500, and the learning goal is 0.04. The performance averaged over 10 networks was listed in Table 4 for comparison.

According to the evaluation results, the TAN based algorithms have better performance than the previous proposed BN based method. A possible reason is that the structure of TAN is learned from the data, while that of BN is based on experts' experiences. The performance of the TAN classifier based algorithm is also comparable to the MLF based algorithm. Although the TAN algorithm has a lower DR, it is superior to MLF based algorithm on FAR. The performance of these two algorithms on MTTD is comparable. However, it should be noted that MLF is much more complicated than TAN classifier in theory. MLF is a black box model (it is difficult to explain the knowledge hidden in it) while TAN classifier is just based on theory of probability and statistics, which has been widely recognized. Additionally, it is much more difficult to train MLF than TAN classifier. The training process of MLF networks needs more data and time. In this study, the training time of the TAN classifier is less than 2 seconds, but the average training time of MLF networks for incident detection is 344.70 seconds. From the analysis above, compared with MLF, the TAN based incident detection algorithm would have wider vista of applications.

6. Conclusions

In this paper, a special form of Bayesian networks, which is called TAN classifier, is used to develop an incident detection algorithm. The Bayesian networks based detection algorithms are developed and evaluated in the previous studies and have shown the superiorities on performance and transferability. However the development of the previous Bayesian networks based algorithms strongly depends on the knowledge of experts. To reduce the dependency on experts' knowledge, the structure of TAN classifier for incident detection is learned from data. The discretization of continuous attributes is processed using an entropy-based method automatically. A simulation dataset on the section of the Ayer Rajah Expressway (AYE) in Singapore is used to demonstrate the procedures of the development of TAN classifier based detection algorithm, including wavelet denoising, normalization, discretization, and structure learning.

The performance of TAN based algorithm is evaluated compared with the MLF based algorithm and the previous developed BN based algorithm using the same AYE data. The experiment results show that the TAN based algorithms have better performances than the previous proposed BN based method. A possible reason is that the structure of TAN is learned from the data, while that of BN is based on experts' experiences. The TAN classifier based algorithm has a similar performance to the MLF based algorithm. However, TAN based algorithm would have wider vista of applications because the theory of TAN classifiers is much less complicated than MLF. It should be found from the experiment that the TAN classifier based algorithm has a significant superiority on the speed of model training and calibration.

Because of the lack of appropriate datasets, the transferability of TAN based algorithm is not evaluated in this paper. In the future, a more comprehensive and detailed evaluation of the TAN based algorithm should be carried out. In the next study, we will also improve our methodology by treating the traffic flow parameters as continuous variables implementing various prior distributions or using the dynamic TAN classifiers.

Abbreviations

Description of Variables Contained in the TAN Classifier

INC: The class variable which indicates the incident state of traffic flow
Q_u: The volume upstream of the incident location
Q_d: The volume downstream of the incident location
V_u: The speed upstream of the incident location
V_d: The occupancy downstream of the incident location
O_u: The occupancy upstream of the incident location
O_d: The occupancy downstream of the incident location
DO: $DO = O_u - O_d$.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the National Natural Science Foundation of China (no. 51608115, no. 51478112), the Natural Science Foundation of Jiangsu Province (BK20150613, BK20150619), the Projects of International Cooperation and Exchange of the National Natural Science Foundation of China (no. 51561135003), the Six Talent Peaks Project in Jiangsu Province (2015-XXRJ-017), the open fund for the Key Laboratory for Traffic and Transportation Security of Jiangsu Province, Jiangsu Province Natural Science in Colleges and Universities Research major projects (7KJA580001), Philosophy and Social Science of Universities in Jiangsu (2016SJB790057), Southeast University Excellent Youth Faculty Research Supporting Project, and the fundamental research funds for the central universities.

References

[1] Usdot, "National conference on traffic incident management: A road map to the future," 2002.

[2] J. A. Lindley, "Urban freeway congestion: quantification of the problem and effectiveness of potential solutions.," *ITE Journal*, vol. 57, no. 1, pp. 27–32, 1987.

[3] H. J. Panyne and S. C. Tignor, "Freeway incident-detection algorithms based on decision trees with states," *Transportation Research Record*, vol. 682, pp. 30–37, 1978.

[4] C. L. Dudek and C. J. Messer, "Incident detection on urban freeway," *Transportation Research Record495*, pp. 12–24, 1974.

[5] M. Levin and G. M. Krause, "Incident detection: a bayesian approach," *Transportation Research Record*, vol. 682, pp. 52–58, 1978.

[6] R. H. Whitson, J. H. Burr, D. R. Drew, and W. R. Mccasland, "Real-time evaluation of freeway quality of traffic service," *Highway Research Record*, vol. 289, pp. 38–50, 1969.

[7] A. R. Cook and D. E. Cleveland, "Detection of freeway capacity-reducing incidents by traffic-stream measurements," *Transportation Research Record*, vol. 495, pp. 1–11, 1974.

[8] L. Aultman-Hall, F. L. Hall, Y. Shi, and B. Lyall, "A catastrophe theory approach to freeway incident detection," in *Proceedings of The The Second International Conference on Applications of Advanced Technologies in Transportation Engineering*, pp. 373–377, New York, NY, USA, 1991.

[9] R. L. Cheu and S. G. Ritchie, *Neural Network Models for Automated Detection of Lane-Blocking Incidents on Freeways*, University of California, Irvine, 1994.

[10] R. L. Cheu and S. G. Ritchie, "Automated detection of lane-blocking freeway incidents using artificial neural networks," *Transportation Research Part C*, vol. 3, no. 6, pp. 371–388, 1995.

[11] K. Zhang and M. A. P. Taylor, "Towards universal freeway incident detection algorithms," *Transportation Research Part C: Emerging Technologies*, vol. 14, no. 2, pp. 68–80, 2006.

[12] K. Zhang and M. A. P. Taylor, "Effective arterial road incident detection: a bayesian network based algorithm," *Transportation Research Part C: Emerging Technologies*, vol. 14, no. 6, pp. 403–417, 2006.

[13] E. Keogh, "Naive bayes classifier. UCR, and Christopher Bishop," in *Pattern Recognition Machine Learning*, Springer, 2006.

[14] G. I. Webb, J. R. Boughton, and Z. Wang, "Not so naive Bayes: aggregating one-dependence estimators," *Machine Learning*, vol. 58, no. 1, pp. 5–24, 2005.

[15] L. Jiang, H. Zhang, Z. Cai, and D. Wang, "Weighted average of one-dependence estimators," *Journal of Experimental Theoretical Artificial Intelligence*, vol. 24, no. 2, pp. 219–230, 2012.

[16] H. Zhang, L. Jiang, and J. Su, "Hidden naive bayes," in *Proceedings of the AAAI*, pp. 919–924, 2005.

[17] L. Jiang, C. Li, S. Wang, and L. Zhang, "Deep feature weighting for naive Bayes and its application to text classification," *Engineering Applications of Artificial Intelligence*, vol. 52, pp. 26–39, 2016.

[18] L. Jiang, D. Wang, and Z. Cai, "Discriminatively weighted naive Bayes and its application in text classification," *International Journal on Artificial Intelligence Tools*, vol. 21, no. 1, Article ID 1250007, 2012.

[19] L. Jiang, Z. Cai, D. Wang, and H. Zhang, "Improving Tree augmented Naive Bayes for class probability estimation," *Knowledge-Based Systems*, vol. 26, pp. 239–245, 2012.

[20] U. B. Kjærulff and A. L. Madsen, *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*, Springer, 2008.

[21] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, 2006.

[22] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Machine Learning*, vol. 29, no. 2-3, pp. 131–163, 1997.

[23] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*, Information Science and Statistics, Springer, Berlin, Germany, 2007.

[24] W. Wang, S. Chen, and G. Qu, "Incident detection algorithm based on partial least squares regression," *Transportation Research Part C: Emerging Technologies*, vol. 16, no. 1, pp. 54–70, 2008.