Transportation Research Part C 117 (2020) 102697

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

Real-time crash prediction on expressways using deep generative models

Qing Cai*, Mohamed Abdel-Aty, Jinghui Yuan, Jaeyoung Lee, Yina Wu

Department of Civil, Environmental and Construction Engineering, University of Central Florida, Orlando, FL 32816, USA

ARTICLE INFO

Keywords:
Real-time crash prediction
Over-sampling
Undersampling
ITS control
Deep convolutional generative adversarial network

ABSTRACT

Real-time crash prediction is essential for proactive traffic safety management. However, developing an accurate prediction model is challenging as the traffic data of crash and non-crash cases are extremely imbalanced. Most of the previous studies undersampled non-crash cases to balance the data, which may not capture the heterogeneity of the full non-crash data. This study aims to use the emerging deep learning method called deep convolutional generative adversarial network (DCGAN) model to fully understand the traffic data leading to crashes. With the full understanding of the traffic data of crashes, the DCGAN model could generate more synthetic data related to crashes to balance the dataset. All non-crash data could be used for developing the prediction models. To capture the correlations between different variables, the data are augmented to 2-D matrix as the input for the DCGAN model. The suggested model is evaluated based on data from expressways and compared to two counterparts: (1) synthetic minority over-sampling technique (SMOTE); (2) random undersampling technique. The results suggest that the DCGAN could better understand the crash data characteristics by generating data with better fit of the real data distribution. Four different crash prediction algorithms (i.e., logistic regression model, support vector machine, artificial neural network, and convolutional neural network) are developed based on each balanced data and totally twelve models were estimated. The results indicate that the convolutional neural network model based on the DCGAN balanced data could provide the best prediction accuracy, validating that the proposed oversampling method could be used for the data balance. Besides, compared to other two models, only the DCGAN-based model could identify the significant effects of speed difference between the upstream and downstream locations which could help guide traffic management strategies. With the prediction model developed based on the balanced data by DCGAN, it is expected that more crashes could be predicted and prevented with more appropriate proactive traffic safety management strategies such as Variable Speed Limits (VSL) and Dynamic Message Signs (DMS).

1. Introduction

Expressways play a vital role in serving traffic demands, especially for the commuting traffic in cities. Despite great attempts having been made, safety issues on expressways are still contains big concerns (Li et al., 2017a). In general, the solutions to reduce crashes could be divided into three categories: (1) planning solutions; (2) engineering solutions; and (3) intelligent transportation systems (ITS) control solutions. For the planning solutions, crashes from a spatial aggregation (such as traffic analysis zone) are

* Corresponding author.
E-mail address: qingcai@knights.ucf.edu (Q. Cai).

https://doi.org/10.1016/j.trc.2020.102697
Received 6 August 2019; Received in revised form 27 May 2020; Accepted 1 June 2020
Available online 15 June 2020
0968-090X/ Published by Elsevier Ltd.

considered to quantify the impact of socioeconomic and demographic characteristics, transportation demands, and network attributes so as to provide countermeasures including education, enforcement, etc. The engineering solutions are to identify the influence of geometric design, lighting with the objective of offering solutions such as adding lighting and increase lane width. The ITS control solutions attempt to provide traffic management strategies such as variable speed limit (VSL) and queue warning by understanding the effects of traffic flow on the crash occurrence. Compared to the planning and engineering solutions, the ITS solutions could enhance the safety within a short term by stabilizing the traffic flow or sending warning messages. Besides, the traffic management could further enhance traffic safety proactively if the crashes could be predicted before they occur. For example, Yu and Abdel-Aty (2014) successfully developed a novel VSL control algorithm to proactively reduce crash risk with the input of predicted crashes risk. Besides, the control strategies considering the connected vehicle technology could further enhance the real-time traffic safety (Li et al., 2017b, 2020a). To be sure, accurate crash prediction should be essential to guarantee the efficiency of the proactive traffic safety management.

Crashes are rare events and the crash and non-crash cases are extremely imbalanced. The previous studies commonly under sampled the non-crash events to balance the training dataset (Shi and Abdel-Aty, 2015; Wang et al., 2019a, 2019b; Yang et al., 2018). Although the method could successfully estimate models and identified reasonable effects of significant variables, most of non-crash cases were not used and their information got lost. Recently, several recent studies adopted over-sampling methods to generate more crash events to balance the data (Basso et al., 2018; Yuan et al., 2019; Li et al., 2020b). Most of them used the synthetic minority over-sampling technique (SMOTE). However, SMOTE is limited to the problems such as generalization and variance issues (He and Garcia, 2008). Meanwhile, the SMOTE method failed to incorporate the correlations of variables during the data oversampling process. Therefore, this study aims to fill this gap and add to the current knowledge by introducing a new data oversampling method based on deep learning. Besides, as the oversampling data would be big data, the deep learning was also used for the crash prediction to improve the accuracy.

2. Literature review

In the recent decades, a wide array of studies has conducted to develop real-time crash prediction models (Basso et al., 2018, 2020; Huang et al., 2020; Roshandel et al., 2015; Wang et al., 2015, 2019a, 2019b; Yu and Abdel-Aty, 2013; Yu et al., 2018, 2020; Yuan et al., 2019). Most of the previous studies predicted crash risk at a 5-min interval and aggregate the traffic data by 5 min. For the development of prediction models, traffic data have extensively used and their impacts on crash occurrence have been investigated. For example, Shi and Abdel-Aty (2015) found that the volume at the upstream has a positive correlation with the crash risk on freeways/expressways. Meanwhile, it was also revealed that the crash risk gets increased if the congestion index at the downstream becomes high. The average speed at the downstream was found to have a significant effect on crash risk (Pande et al., 2011). Xu et al. (2013a, 2013b) concluded that the speed diference between the upstream and downstream locations is significantly correlated to the crash risk. The high speed diference could lead to crash occurrence. Further, Yang et al (2018) found that higher speed standard deviation could increase the crash risk. The variables should be included for the crash risk prediction and they could also help select the appropriate traffic safety management countermeasures.

In general, the methods for real-time crash risk prediction could be divided into two categories: statistical and machine learning methods. The statistical methods include simple or match-case logistic model, log-linear model, Bayesian logistic model, etc. (Abdel-Aty et al., 2004; Wang et al., 2015, 2019a, 2020; Yuan et al., 2018). These models could identify contributing factors in a more interpretable way. On the other hand, machine learning methods such as support vector machine (Yu and Abdel-Aty, 2013; Basso et al., 2018), neural network (Pande et al., 2011), and random forest (Lin et al., 2015) could achieve high prediction accuracy. Recently, with the rapid development of deep learning, some studies started to use deep learning to predict crash risk and found that the deep learning could significantly improve the prediction performance. Huang et al. (2020) applied the convolutional neural network (CNN) with drop-out operation to predict crash risk on interstate roads and found that the CNN model performs better than shallow neural network models. Bao et al. (2019) used the CNN model to conduct the citywide short-term crash risk prediction. It was suggested that the CNN modeling structure could capture locally spatial correlation. Yuan et al. (2019) utilized long short-term memory neural network (LSTM) model to predict crash risk for arterials in real time. Much better sensitivity performance could be achieved compared to the conditional logistic model. Besides, Li et al. (2020b) and Yuan et al. (2020) combined LSTM and CNN to address temporal and spatial relationships for the crash risk prediction. The CNN was used to handle the spatial correlation between variables at different locations.

Given the fact that the traffic data of crash and non-crash cases are extremely imbalanced, it is necessary to balance the data before developing crash prediction models. Compared to the development of different analysis models, the methods to balance the dataset seem not gaining much attention. Most of the previous studies applied the undersampling methods to balance the dataset. Matched case-control design has been extensively used to select non-crash data. As only a small number of non-crash data is used, the undersampling method may lose some important information of data associated to non-crash events. Only several recent studies adopted over-sampling methods to generate more crash events to balance the data. All of them used the synthetic minority over-sampling technique (SMOTE). For example, Basso et al. (2018) used the SMOTE method to oversample the crash data on freeway segments and calibrated the crash prediction algorithm. The calibrated algorithm was evaluated with the real-world imbalanced dataset and found that the algorithms based on SMOTE balanced dataset have better prediction performance. Noteworthy, in the previous studies, real-time crash risk prediction models were evaluated based on artificially balanced test dataset, and the accuracy could not represent the accuracy in practice. Yuan et al. (2019) and Li et al. (2020b) extended SMOTE to arterials and validated the advantage of the SMOTE oversampling approach. The SMOTE method was also adopted in other recent real-time crash risk prediction

studies (Ke et al., 2019; Elamrani Abou Elassad et al., 2020). Schlögl et al. (2019) used a combination of SMOTE oversampling and maximum dissimilarity undersampling to balance the training dataset. The balanced data were used to compare different machine learning algorithms. Schlögl (2020) introduced a balanced bagging approach to down sample bootstrap training samples. Despite the increasing popularity of using SMOTE to balance data, SMOTE is limited to the problems such as generalization and variance issues (He and Garcia, 2008). Meanwhile, the traffic status at the upstream and downstream should be highly correlated. However, the data generation process of the SMOTE method fails to incorporate such correlations among variables.

Recently, Goodfellow et al. (2014) proposed a deep neural network called generative adversarial network (GAN) to create generative models. A generative model is a powerful way of learning any kind of data distribution using unsupervised learning and it has achieved tremendous success in the recent years. Since the invention of GAN, it has been adopted to help enlarge original dataset and balance label distribution from data augmentation in different fields such as image processing (Zhu et al., 2017) and bank fraud detection (Fiore et al., 2017). It has indicated that the GAN is a general, flexible, and powerful generative deep learning models for over sampling data. Schlögl et al. (2019) suggested that GAN could be a good alternative to oversample crash data to get the training data balanced. However, to our best knowledge, no study has used GAN for over sampling traffic data of crash events to balance the crash prediction data.

In summary, this study attempts to introduce a new method to balance the dataset for real-time traffic crash prediction. To specify, the study contributes the literature for real-time crash prediction along three directions: (1) introducing the GAN model to generate traffic data associate with crashes for the data balance; and (2) using the CNN model to incorporate the correlations between variables; (3) conducting extensive comparison studies for oversampling and undersampling data based on different types of crash prediction models. The follow paper is organized as follows: Section 3 described the collected data. Section 4 presented the GAN oversampling method as well as other data balance methods. The crash risk prediction models were also presented in this section. Section 5 summarized the model performance and modeling results. Finally, the last section summarizes and concludes this paper.

3. Data preparation

The expressway SR 408 in Orlando is selected in this study. SR 408 travels through downtown Orlando and serves as one of major expressways for commuter traffic. SR 408 is 21.4 miles long per each direction, with a total 110 MVDS detectors have been installed. Hence, the average distance between adjacent detectors are around 0.5 miles. The MVDS detectors could detect spot speed, volume, and occupancy of each lane in real time, which enable traffic operators to manage the system and take traffic management strategies in real time. In this study, the MVDS data in 2017 were collected and cleaned to eliminate the unreasonable traffic data such as negative speed and volumes. Based on the MVDS detector data, other indicators such as average speed and standard deviation of speed could be computed as well. Data were aggregated into 5-minute intervals for analysis.

During 2017, a total of 625 crashes occurred on SR 408. For each crash, traffic data of 5–10 min prior to the crash from two upstream and two downstream MVDS detectors (see Fig. 1) closest to the crash location are collected and aggregated. Meanwhile, other traffic data at the same location six hours and six hours before and after crash were eliminated to avoid the noise caused by the crashes. At each location × (i.e., U1, U2, D1, and D2), six traffic parameters including logarithm of volume (volume_x), logarithm of average, standard deviation, and coefficient of variation of the speed (avg_speed_x, std_speed_x, and cv_speed_x), speed difference between inner and outer lanes (speed_diff_x), and congestion index (CI_x) were extracted. In this study, the congestion index is calculated as:

$$Congestionindex\ \ (CI) = \begin{cases} \frac{speedlimit - actualspeed}{speedlimit} & if \quad CI > 0 \\ 0 & if \quad CI \le 0 \end{cases}$$

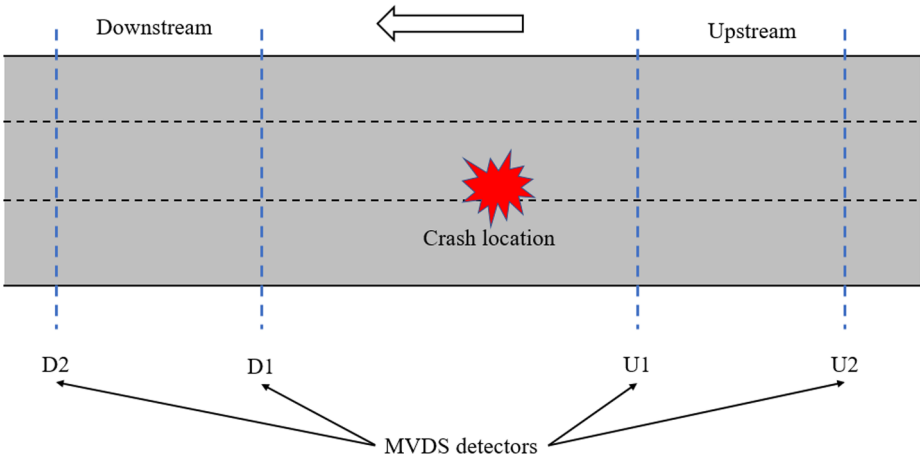

Fig. 1. Locations of Crash and MVDS detectors.

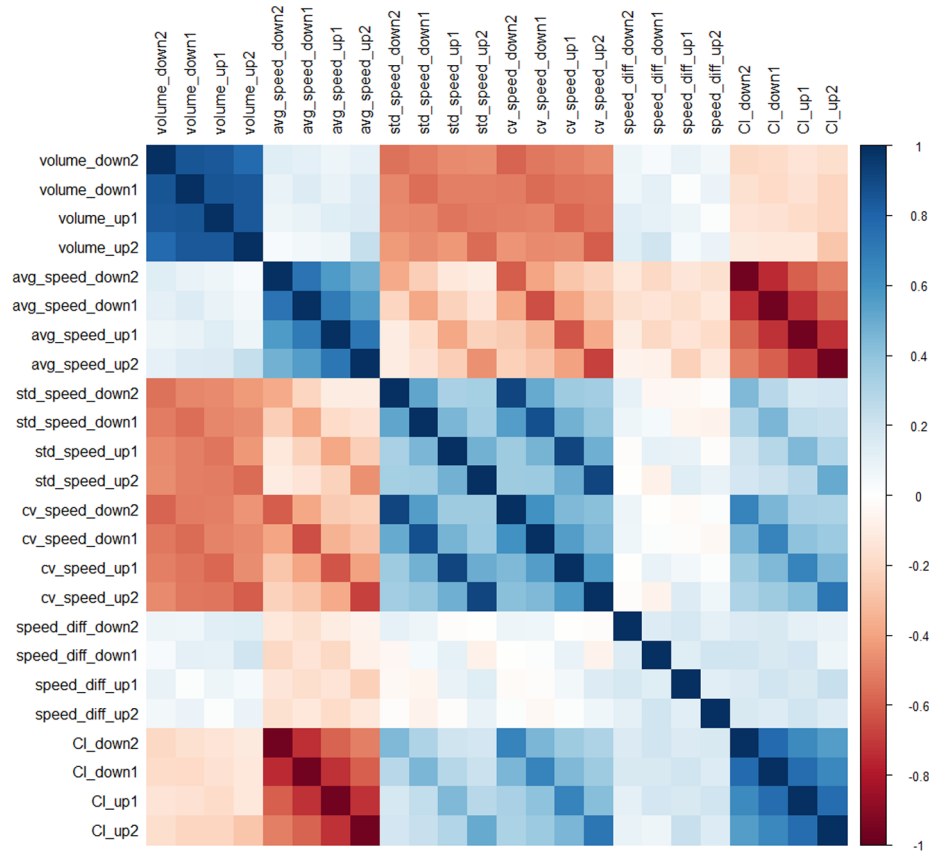

Fig. 2. Variable correlation.

As suggestion in the previous study (Shi and Abdel-Aty, 2015), CI is a flexible and consistent term to reflect the congestion intensity.

By combing detectors at four consecutive locations as a group, totally 6,750,072 traffic data could be obtained. Hence, the ratio of crash and non-crash cases is 625: 6,749,447, which is around 1: 11,000. Hence, the data is extremely imbalanced and make it impossible to develop a prediction model directly. In the following section, the deep convolutional generative adversarial network is introduced to over sample the traffic data of crash cases. Meanwhile, traditional methods including SMOTE over-sampling and random undersampling are also presented.

4. Method

4.1. Traffic data analysis and augmentation

The correlation of the 24 aggregated variables is displayed in Fig. 2. The blue color indicates the positive correlations while red indicates the negative correlations. The darker color suggests higher correlations. It is shown that the same variables except speed difference from the four locations are highly correlated. Meanwhile, some variables at the same location have large correlations. For example, the average speed at D1 is positively associated to the average speed at U1, and negatively correlated to the congestion index at D1. The correlation should be considered when generating the crash data.

4.2. Deep convolutional GAN-based data balance

Generative Adversarial Network (GAN) (Goodfellow et al., 2014) consists of two networks (a generator G network and a discriminator D) that contest with each other. The objective of the generator is to produce data, which could confuse the discriminator. On the other hand, the objective of the discriminator is to distinguish the instances coming from the generator and the instances from the original dataset. If the discriminator could identify the instances from generator easily, relative to the discrimination ability, the generator is producing data with low quality. With the feedback from the discrimination, the generator is forced to increase its performance. The competition between the generator and discriminator that can be formalized as a minimax function:

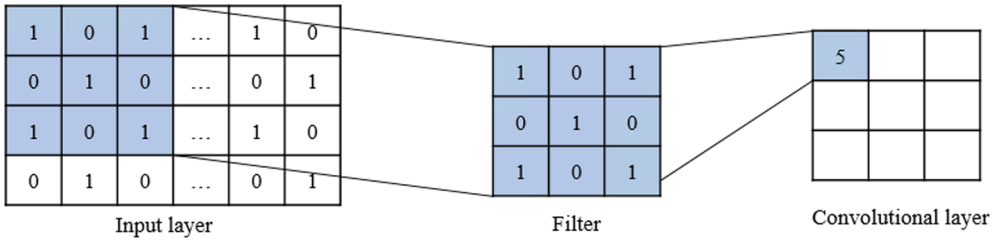

Fig. 3. Example of convolutional feature extraction.

$min_G max_D(D, G) = E_{x p_{(x)}}[\log D(x)] + E_{z p_z(z)}[\log(1 - D(G(z)))]$where $p_{(x)}$ is the training data distribution, $p_z(z)$ is the prior distribution of the generative network, and z is a noise vector sampled from the model distribution $p_z(z)$ such as the Gaussian or uniform distribution. In other words, the goal of generator is to minimize the differentiation between real and synthetic data, whereas the discriminator aims to maximize the probability of distinguishing real data from synthetic ones.

The data generation should incorporate the high correlations between variables. For example, when generating the average speed at D1, the corresponding average speed at U1 and congestion index at D1 should be generated. Hence, the 1-D traffic data is transformed into a 2-D (4 × 6) data matrix and convolutional neural network (CNN) is adopted to capture the correlations between variables. It has been proven that the CNN model has shown its powerful ability to hierarchically capture the spatial correlation information (Cai et al., 2019; LeCun et al., 1998; Wu et al., 2018a). The 2-D data matrix is as follows:

$$\begin{bmatrix} volume\_down2, & avg\_speed\_down2, & std\_speed\_down2, & cv\_speed\_down2, & speed\_diff\_down2, & CI\_down2, \\ volume\_down1, & avg\_speed\_down1, & std\_speed\_down1, & cv\_speed\_down1, & speed\_diff\_down1, & CI\_down1, \\ volume\_up1, & avg\_speed\_up1, & std\_speed\_up1, & cv\_speed\_up1, & speed\_diff\_up1, & CI\_up1, \\ volume\_up2, & avg\_speed\_up2, & std\_speed\_up2, & cv\_speed\_up2, & speed\_diff\_up2, & CI\_up2, \end{bmatrix}$$

The same row contains the different variables at the same location while the same column contains the same variables at different locations. For example, the first column contains volumes at locations at downstream 2, downstream 1, upstream 1, and upstream 2. As the real input data is in 2 dimensions, the convolutional neural network (CNN) is adopted in the discriminator. Recently, CNN is widely used in image recognition applications and proven to be a powerful tool to understand the spatial interaction in the local matrix units. Recently, it has been also introduced for traffic crash risk prediction (Bao et al., 2019; Cai et al., 2019; Huang et al., 2020). CNN learns the local correlation through convolution. By using small filters, the input matrix layer could be projected to multiple convolutional layers, which could subtract the relations among local matrixes. As shown in Fig. 3, a filter is applied on the input matrix and the convolved feature is calculated by element-wise multiplication and summation (e.g., $1 \times 1 + 0 \times 0 + 1 \times 1 + 0 \times 0 + 1 \times 1 + 0 \times 0 + 1 \times 1 + 0 \times 0 + 1 \times 1 = 5$). The weights on the filter are shared with any locations of the matrix. Several convolutional layers could be generated, connected with several fully connected layers. The CNN could be applied in the discriminator for a binary classification to determine whether the data is real or generated.

On the other hand, the generator applies the transposed convolution (deconvolution) process to generate the data matrix with the random noise. The model structure of the DCGAN model is presented in Fig. 4. The training process of generator is opposite to the training process of the discriminator. Firstly, a fully connected layer with random noise is generated, connected to several other fully connected layers. Then, the deconvolution process is applied to convert the fully connected layers to the convolutional layers, which

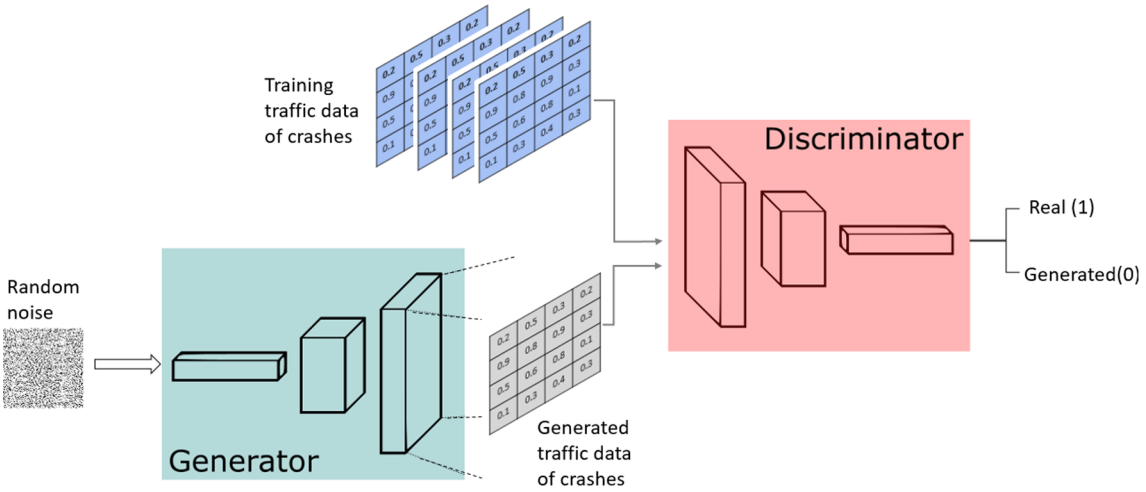

Fig. 4. The architecture of the DCGAN for traffic data of crashes generation.

Table 1
DCGAN model architecture.

Generator CNN architecture			Discriminator CNN architecture		
#Layer	Layer type	Layer description	#Layer	Layer type	Layer description
1	Fully connected + BN +ReLU +Dropout	#Neuron = 1280 Output size = 1280 × 1	1	Convolution +BN +LeakyReLU(0.2) +Dropout	Filter size = 3 × 3 Stride = 1, Padding = 2 Output size = 8 × 6 × 8
2	Fully connected + BN + ReLU + Dropout	#Neuron = 3840 Output size = 3840 × 1	2	Convolution +BN +LeakyReLU(0.2) +Dropout	Filter size = 3 × 3 Stride = 1, Padding = 2 Output size = 16 × 8 × 10
3	DeConvolution + BN +ReLU +Dropout	Filter size = 3 × 3 Stride = 1, Padding = 2 Output size = 16 × 8 × 10	3	Convolution +BN +LeakyReLU(0.2) +Dropout	Filter size = 3 × 3 Stride = 1, Padding = 2 Output size = 32 × 10 × 12
4	DeConvolution + BN +ReLU +Dropout	Filter size = 3 × 3 Stride = 1, Padding = 2 Output size = 8 × 6 × 8	4	Fully connected + LeakyReLU(0.2) +Dropout	#Neuron = 256 Output size = 256 × 1
5	DeConvolution + BN +ReLU +Dropout +tanh	Filter size = 3 × 3 Stride = 1, Padding = 2 Output size = 1 × 4 × 6	5	Fully connected + Sigmoid	#Neuron = 1 Output size = 1

could output data matrixes. In this study, the output matrix contains 24 variables with the dimension of (4 × 6), i.e., the 24 variables could be generated at the same time and the correlations between variables are incorporated.

Table 1 summarizes the DCGAN model architecture developed in this study. Both the generator and discriminator network have five layers, of which 3 layers are (de)convolution layers and 2 layers are fully connected layers. For the convolution in the discriminator, the relatively small filter size (3 × 3) was used as the input matrix dimension is only 4 × 6. The Stride = 1 and Padding = 2 were used to ensure that the dimension of matrix increased gradually. As the generator is a deconvolution process, the same structure was used. According to the suggestion by Inkawhich (2020), LeakyRelu was used for the discriminator and Relu was used for the generator. Batch Normalization (BN) and dropout techniques were used to better train the model.

A vector of 256 noise points with normal distribution (mean = 0, standard deviation = 0.2) is used as the input for generator network. On the other hand, a matrix of 4 × 6 is the input of the discriminator network. The initial matrix follows a normal distribution (mean = 0, standard deviation = 1) so that most initial data are between −1 and 1, which is the range of the input normalized traffic data. All model weights were randomly initialized from a normal distribution with mean = 0 and standard deviation = 0.02, as suggested by Li et al. (2020b). With the trained DCGAN model, traffic data of crashes could be generated and added to balance the crash and non-crash cases.

Training DCGAN is known to not be an easy task (Arjovsky and Bottou, 2017). To get promising synthetic data, the following steps were adopted when tuning the network parameters according to the previous studies (Isola et al., 2017) and characteristics of traffic data:

- Use a hyperbolic tangent function (tanh) instead of a Sigmoid function as the last layer of the generator output. The tangent function could transform data to the range from −1 and 1 with the distribution peak at zero.
- Normalize the input traffic data between −1 and 1. The following function is used:

$$x_{tranform} = \frac{2(x - x_{min})}{x_{max} - x_{min}} - 1$$

where x is the original input data and $x_{tranform}$is the transformed data, x_{min} and x_{max} are the minimum and maximum values of x. This transformation method has been widely used in the previous studies or practice related to GANs. However, if the distribution peak of the input data is not equal to the mean of the input data, the distribution peak of transformed data will not be at zero. Thus, this study is suggesting to further transform the $x_{tranform}$ transformed by:

$$x^{'}_{transform} = \tanh(x_{tranform} - x^{peak}_{transform})$$

where $x^{'}_{transform}$ is the data after second transformation and $x^{peak}_{transform}$ is the density peak point of data $x_{tranform}$. After the second transformation, the data is between −1 and 1 with the distribution peak at zero. To our best knowledge, no studies have adopted the second transformation. It should also be noted that the inverse transformation is used to transform the generated value to the traffic data.

- Change the loss function to optimize G by maximizing log instead of minimizing $\log(1 - D(G(z)))$ to avoid early vanishing

gradients (Isola et al., 2017).
- Generate noise from a normal distribution rather than a uniform distribution.
- Smooth label. Instead of using label 1 for real data and 0 for synthetic data, a random number between 0.7 and 1.2 is assigned to the real data while the synthetic data is labelled with a value from 0 to 0.3.
- Use different optimizer for generator and discriminator. This study used SGD for discriminator and ADAM for generator, to obtain the most stable training results.
- Construct different mini-batches for real and fake, i.e. each mini-batch needs to contain only all real images or all generated images.

Since the full training data could be loaded into the GPU once, the batch size of 1 was used. Different number of epochs (i.e., 20, 50, 100, 200, 300, 400, 500) were tested, and 200 was selected finally. Heusel et al. (2017) provided a mathematical proof of convergence to Nash equilibrium and showed that using different learning rates for the generator and discriminator could achieve promising training performance. The rates of 0.0005 and 0.0001 were used for the discriminator and generator in this study, separately. Theoretically, the discriminator converges to 0.5 in the GAN network, which indicates that it is impossible to distinguish between input real data and synthetic data because they are samples of the same distribution (Heusel et al., 2017). The discriminator loss in this study converged around 0.53 (close to 0.5), indicating the synthetic data and real data were similar and the discriminator is difficult to distinguish them.

4.3. Other data balancing methods

4.3.1. SMOTE over-sampling

The Synthetic Minority Over-sampling Technique (SMOTE) is a traditional over-sampling method (Chawla et al., 2002). SMOTE creates vectors between each example of the minority datasets (traffic data of crashes) and its k-nearest neighbors (in the same dataset). In this study, k was set to be 5, which is consistent with Chawla et al. (2002) and Yuan et al. (2019). Then, a new example data vector is added to the dataset by multiplying each vector by a random constant between 0 and 1. The process is repeated until reaching the target sampling rate. Different from the DCGAN, SMOTE requires the input data to be 1-D vector. Hence, SMOTE may not able to well incorporate the correlations between different variables.

4.3.2. Random undersampling

The undersampling technique is to under sample the majority class (traffic data of non-crash cases). The random undersampling technique is to under sample the majority class (non-crash data) randomly and uniformly. It is expected that undersampling data could represent the distribution of the majority data. It should be noted that the minority data (i.e., traffic data for crash events) will not change.

4.4. Crash prediction model

With the over-sampling and undersampling techniques, the crash and non-crash data could get balanced. To conduct an extensive comparison study, four different models were introduced including logistic regression, support vector machine (SVM), artificial neural network (ANN), and convolutional neural network (CNN) models. The four models are specified in the following subsections.

(1) Logistic regression model

The logistic regression model which has been widely used in the real-time safety studies (Shi and Abdel-Aty, 2015; Yuan et al., 2019). The dependent variable is a binary indicator of crash occurrence, with the probability p for crash case ($y = 1$) and $1 - p$ for non-crash case ($y = 0$). The logit model could be expressed as:

$$y \sim Bernoulli(p)$$

$$logit(p) = \alpha + \boldsymbol{\beta x}$$

where α is the intercept term, $\boldsymbol{x}$ is the vector of independent variables, and $\boldsymbol{\beta}$ is the vector of corresponding coefficients.

(2) Support Vector Machine (SVM)

Support vector machine was originally proposed based on statistical learning theory and the structural risk minimization. The model is to find a separating hyperplane by minimizing the distance of misclassified points to the decision boundary. The SVM model could be used to solve binary classification problem and it has been adopted to predict crashes vs non crashes (Yu and Abdel-Aty, 2013; Basso et al., 2018). The SVM tries to find the function $f(x, \alpha_0) \in f(x, \alpha)$, which best approximates the unknown function $y = f(x)$. The hyperplanes could be written as:

$$f(x, \alpha) = (\boldsymbol{w_a} \cdot \boldsymbol{x}) + b$$

(3) Artificial Neural Network (ANN)

The typical ANN model contains multiple neurons in different layers: an input layer, output layer, and several hidden layers. For the crash classification, the output is the labelled data. The connection between layers are formed with learnable weights and bias. By tuning the weights and bias on each connection between neurons, the output approaches the truth. To account for non-linearity, activation functions are used to normalize the output of input and hidden layers. Normally, rectified linear unit (ReLU) is used:

$$\text{ReLU:}\quad f(x) = \begin{cases} 0, & for \quad x < 0 \\ x, & for \quad x \geq 0 \end{cases}$$

The loss function is used to compute the difference between network output and the ground truth. For the binary classification problem, the binary cross entropy loss function is preferred.

(4) Convolutional Neural Network (CNN)

As discussed above, there are correlations between different variables at the same location and the same variables at different locations. Such correlation is not considered in the above models. Hence, the CNN model could be also adopted for the crash prediction. Different from the ANN model, the input data for the CNN model is similar to images. A small filter is applied that slides over the large image, to capture the local features. Finally, several fully connected layers are used to connect the filtered output. The output layer is same as the ANN model and the similar procedures are used to tune the fully connected layer and output layer parameters.

5. Results and discussions

5.1. Data sampling

The full dataset with 6,750,072 observations (625 crash cases and 6,749,447 non-crash cases) was divided into training dataset (70%) and test dataset (30%). The over-sampling methods and undersampling method were applied on the training dataset to get the balance dataset for developing the prediction model, whereas the test dataset was still the originally imbalanced data. Hence, the developed model could be evaluated based on the fully test dataset. Among the previous studies, the ratio of 1:4 has been the most commonly used (Shi et al., 2015; Wang et al., 2015, 2019b; Yuan et al., 2019; Li et al., 2020b). A study conducted by Roshandel et al (2015) concluded that a ratio of 1:4 is recommended since the statistical power generally does not increase significantly beyond that. Besides, Zheng et al (2010) proved that the coefficient of variables become stable when the ratio reaches 1:4. Hence, the ratio of 1:4 was adopted to balance the training dataset over the three sampling methods.

The data size is summarized in Table 2. The original training dataset has a total of 4,725,049 data, of which 437 data are crash-related and 4,724,612 data are for non-crash cases. Based on the 437 crash-related data, the DCGAN and SMOTE generated 944,487 (944,924-437) synthetic crash-related data. The synthetic and real crash-related traffic data were combined for developing the prediction model. On the other hand, the random undersampling method selected 2185 non-crash traffic data from the original training dataset. For different prediction models, the same test dataset which has 188 crash cases and 2,024,835 non-crash cases was used.

Fig. 5 illustrates examples of synthetic data distributions for the variables ‘volume_down2’ and ‘speed_diff_up1’. It could be found that the DCGAN model could better capture the data characteristics and provide well fit of traffic data distribution associated to crashes. This also validated that the DCGAN model got converged since the generator could provide synthetic data close to the real data distribution. The SMOTE method could also generate the data with the same range as the original data, while the DCGAN method could provide a slightly smoother distribution. On the other hand, the random undersampling method could do well sampling if the original data is normally distributed (i.e., volume_down2). Otherwise, the undersampling data could not well fit the distribution of the original non-crash traffic data (i.e., speed_diff_up1).

The above 24 aggregated variables indicate traffic statuses for different locations separately, which don’t reflect the relation between the upstream and downstream locations. The previous studies suggested that the speed difference between upstream and downstream locations was significantly correlated with the crash risk (Lee et al., 2003; Xu et al., 2013a, 2013b; Wu et al., 2018b). Hence, in addition to the 24 aggregated variables, three variables (i.e., speed_diff_up2up1, speed_diff_up1down1, and speed_diff_down1down2) indicating the speed difference between the upstream and downstream locations were calculated. For example, “speed_diff_up2up1” indicates the speed difference between detectors at upstream2 and upstream 1 locations. The speed difference is

Table 2
Summary of data size.

Data type	Total dataset	Test dataset	Original training dataset	Over-sampling training dataset	Undersampling training dataset
Crash	625	188	437	944,924	437
Non-crash	6,749,447	2,024,835	4,724,612	4,724,612	2185
Total	6,750,072	2,025,023	4,725,049	5,669,536	2622

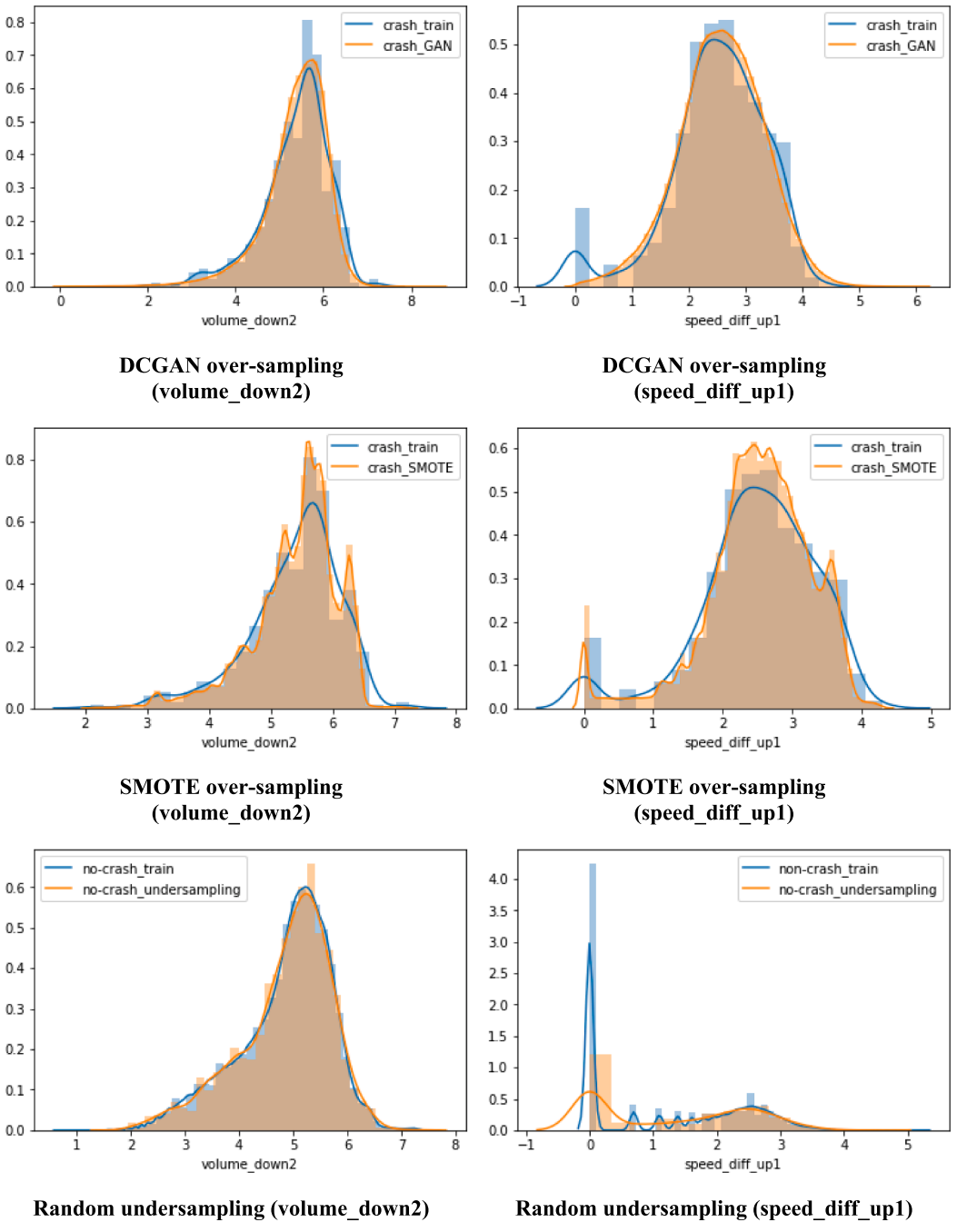

Fig. 5. Illustration of synthetic crash data.

assigned to be 0 for the negative value. As the speed difference is calculated by using the average speeds at two locations, it is not necessary to include the speed difference in the over-sampling or undersampling process. Table 3 presents the statistical description of all data for different datasets.

5.2. Model comparison

In this study, from the 6,750,072 observations (625 crash cases and 6,749,447 non-crash cases), 70% of the observations were selected for training crash prediction models and 30% were used for validating the trained models. Before the model development, the training data was balanced by three different methods (DCGAN oversampling, SMOTE oversampling, and random undersampling). The overall model estimation process involved estimating twelve models – 4 model types (logistic regression, SVM, ANN,

Table 3
Summary of statistical description.

Variables	Oversampling-GAN				Oversampling-SMOTE				Undersampling				Test			
	mean	std.dev	min	max	mean	std.dev	min	max	mean	std.dev	min	max	mean	std.dev	min	max
volume_down2	4.97	0.85	0	10.47	4.97	0.85	0.69	7.57	4.96	0.85	2.08	7.31	4.87	0.85	0.69	7.47
avg_speed_down2	4.08	0.25	0.10	5.35	4.07	0.26	0.10	4.60	4.07	0.27	2.45	4.36	4.11	0.20	0.10	4.61
std_speed_down2	2.14	0.79	0	9.88	2.14	0.77	0	4.55	2.14	0.77	0.42	3.73	2.17	0.80	0	4.40
speed_diff_down2	1.49	1.23	0	6.36	1.49	1.23	0	5.36	1.51	1.22	0.00	4.13	1.25	1.21	0	5.02
cv_speed_down2	0.18	0.18	0	6.66	0.18	0.18	0	6.66	0.18	0.17	0.01	0.94	0.18	0.19	0	6.65
CI_down2	0.15	0.15	0	1.00	0.15	0.16	0	1.00	0.15	0.16	0.00	0.85	0.13	0.13	0	1.00
volume_down1	4.99	0.85	0.01	8.26	4.99	0.85	0.69	7.57	4.98	0.87	1.61	7.22	4.87	0.85	1.10	7.56
avg_speed_down1	4.08	0.25	0.10	4.83	4.07	0.26	0.10	4.62	4.06	0.28	1.53	4.36	4.11	0.20	0.10	4.61
std_speed_down1	2.13	0.77	0	8.63	2.13	0.76	0	4.44	2.16	0.78	0.00	4.22	2.17	0.79	0	4.55
speed_diff_down1	1.50	1.24	0	6.04	1.49	1.24	0	4.83	1.55	1.24	0.00	4.25	1.25	1.21	0	4.85
cv_speed_down1	0.17	0.17	0	6.66	0.18	0.18	0	6.66	0.18	0.19	0.00	2.97	0.18	0.18	0	6.66
CI_down1	0.15	0.15	0	1.00	0.15	0.16	0	1.00	0.16	0.17	0.00	0.95	0.12	0.13	0	1.00
volume_up1	5.00	0.85	0.03	8.73	5.00	0.85	0.69	7.57	5.00	0.86	2.08	7.25	4.88	0.85	0.69	7.55
avg_speed_up1	4.08	0.25	0.47	4.87	4.07	0.26	0.47	4.62	4.07	0.26	2.46	4.35	4.11	0.20	1.25	4.61
std_speed_up1	2.15	0.78	0	7.96	2.15	0.76	0	4.26	2.16	0.77	0.42	3.70	2.18	0.79	0	4.08
speed_diff_up1	1.49	1.27	0	7.74	1.49	1.25	0	5.17	1.53	1.25	0	4.30	1.23	1.22	0	4.80
cv_speed_up1	0.18	0.17	0	2.31	0.18	0.17	0	2.31	0.18	0.17	0.01	1.01	0.18	0.18	0	1.46
CI_up1	0.15	0.16	0	1.00	0.15	0.16	0	0.99	0.15	0.16	0	0.84	0.13	0.13	0	0.96
volume_up2	4.97	0.89	0.02	9.03	4.97	0.88	0.69	7.57	4.95	0.90	1.39	7.25	4.85	0.88	0.69	7.51
avg_speed_up2	4.07	0.24	0.47	4.83	4.07	0.25	0.47	4.62	4.07	0.26	2.47	4.34	4.10	0.21	0.88	4.61
std_speed_up2	2.17	0.80	0	10.69	2.16	0.77	0	4.26	2.17	0.78	0	3.65	2.21	0.80	0	4.08
speed_diff_up2	1.47	1.25	0	8.23	1.47	1.25	0	5.32	1.51	1.25	0	4.13	1.23	1.23	0	5.24
cv_speed_up2	0.18	0.18	0	2.31	0.18	0.18	0	2.31	0.18	0.18	0	1.03	0.19	0.19	0	1.70
CI_up2	0.15	0.16	0	1.00	0.15	0.15	0	0.99	0.15	0.16	0	0.84	0.13	0.13	0	0.98
speeddiff_down1up1	3.11	5.43	0	87.77	2.92	5.25	0	71.10	3.19	5.59	0	46.80	2.73	4.83	0	70.88
speeddiff_down2down1	3.02	5.18	0	96.19	2.91	5.18	0	71.10	2.80	5.13	0	52.85	2.85	4.93	0	70.88
speeddiff_up1up2	2.98	5.01	0	74.49	2.96	5.18	0	60.47	3.10	5.38	0	50.33	2.72	4.77	0	71.80

and CNN models) for three different balanced training datasets. To evaluate the model performance, AUC which indicates the area under the receiver operating characteristics (ROC) curve was adopted in this study. The ROC curve reflects the overall performance of a classification model at all classification thresholds. Besides, the sensitivity and specificity were adopted to measure the performance of the models for crash and non-crash cases, respectively.

The different algorithms were tuned to obtain the best performance metrics. For the logistic regression, the Pearson correlation of the 27 variables were checked and a threshold of 0.7 was used for selecting the significant variables. In the previous studies, different thresholds from 0.3 to 0.95 were used (Basso et al., 2018; Wang et al., 2015, 2019b; Yuan et al., 2019). The threshold of 0.7 was adopted to include more important variables and ensure the variables included in the models not to affect the parameter sign. Besides, the variable importance (decrease in Gini impurity index) was generated by using the random forest algorithm. If two variables were highly correlated, the variable with higher importance was included. In addition, if the parameter sign of a variable got changed by adding a new variable, the less important variable would be excluded. The modeling results about the identified variables are presented and discussed in the following subsection. For the SVM model, the package 'sklearn' in the Python program was used. The 'modified huber' loss function which could output the probability of crash occurrence was used. The 'modified huber' loss could be calculated as follows (Zhang et al., 2002):

$$loss(p, y) = \begin{cases} 0, & py \geq 1 \\ (1 - py)^2, & py \in [-1, 1) \\ -4py, & py < -1 \end{cases}$$

where p is the predicted probability and y is the true label. The probability p could be estimated by:

$$p = \begin{cases} 1, & f(x) \geq 1 \\ (f(x) + 1)/2, & f(x) \in [-1, 1) \\ 0, & f(x) < -1 \end{cases}$$

f(x) is the function with the independent variables. It was indicated that "modified huber" is a smooth loss which brings tolerance to outliers as well as probability estimates (scilit-learn, 2020).

Different number of iterations (i.e., 10, 100, 1000, 10,000, 100,000) were tested for different dataset. The optimal number of iterations for DCGAN, SMOTE, undersampling training datasets are 1000, 1000, and 10,000, respectively. The two neural network models (ANN and CNN) were tuned based on Pytorch using NVIDIA RTX 2080 Ti 11G GPU. The hyperparameters were carefully tuned to achieve the best results. The tuning process and results are summarized in Table 4. As the undersampling training dataset contains 2622 observations, it is not necessary to use batch training. Instead, all training data were included for each update of

Table 4
Hyperparameters tuning.

Hyperparameter			Learning rate	Epoch number	Batch size
Range			0.001, 0.01, 0.02, 0.05, 0.1	1, 5, 10, 15, 20, 25, 30	1000, 10,000, 100,000,
Value	ANN	DCGAN	0.01	10	10,000
		SMOTE	0.01	15	10,000
		Undersampling	0.02	30	–
	CNN	DCGAN	0.01	10	10,000
		SMOTE	0.01	20	10,000
		Undersampling	0.05	30	–

Table 5
Model performance for training datasets.

Measure	Dataset	Logistic model		SVM model		ANN model		CNN model	
		Synthetic crash	Real crash	Synthetic crash	Real crash	Synthetic crash	Real crash	Synthetic crash	Real crash
Sensitivity	DCGAN	0.787	0.760	0.905	0.844	0.899	0.837	0.908	0.851
	SMOTE	0.762	0.757	0.887	0.867	0.884	0.878	0.885	0.829
	UNDER SAMPLING	–	0.759	–	0.875	–	0.627	–	0.510
Specificity	DCGAN	0.759		0.905		0.899		0.908	
	SMOTE	0.764		0.887		0.884		0.865	
	UNDER SAMPLING	0.764		0.874		0.628		0.513	
AUC	DCGAN	0.867		0.959		0.959		0.965	
	SMOTE	0.865		0.952		0.954		0.963	
	UNDER SAMPLING	0.861		0.841		0.644		0.513	

Table 6
Model performance for test datasets.

Measure	Dataset	Logistic model	SVM model	ANN model	CNN model
Sensitivity	DCGAN	0.763	0.846	0.872	0.888
	SMOTE	0.762	0.857	0.838	0.854
	UNDER SAMPLING	0.756	0.628	0.468	0.505
Specificity	DCGAN	0.761	0.800	0.884	0.907
	SMOTE	0.761	0.851	0.848	0.854
	UNDER SAMPLING	0.759	0.848	0.764	0.459
AUC	DCGAN	0.859	0.936	0.948	0.956
	SMOTE	0.858	0.944	0.952	0.953
	UNDER SAMPLING	0.855	0.832	0.647	0.503

parameters.

The performances of the training and test datasets were summarized in Tables 5 and 6. The tables present the AUC, sensitivity, specificity for training and test datasets. To calculate specific values for the performance values, the threshold needs to be determined for each model. In this study, the threshold value was selected as the point where the smallest difference between sensitivity and specificity is obtained. This threshold selection method has been widely used in the previous studies of real-time crash risk prediction (Shi et al., 2015; Yuan et al., 2019). Based on this determined threshold, the three performance measures could be calculated for each model. It should be noted that the AUC was used for the hyperparameters turning process and a threshold of 0.005 was applied to determine if there is a significant difference. Besides, as the DCGAN and SMOTE added synthetic crashes to the real crash to balance the crash and non-crash cases. The sensitivity was calculated separately for the synthetic and real crash cases in the training dataset. Several observations can be made from the results. First, the logistic regression models developed based on the three training datasets have the similar performance while the model based on the DCGAN data is slightly better but not significantly. Second, among the models developed based on the DCGAN and SMOTE (oversampling) data, the three machine learning methods could achieve significantly better results compared to the logistic regression model. The results validated that the machine learning methods outperform conventional statistical models for learning the large non-linear data. Compared to the real crashes, the generated synthetic crashes have higher sensitivities in all models based on the oversampling data. Although the synthetic crashes are based on the real crashes, there still exists difference between them. Since the synthetic crashes are much more than the real crashes in the training dataset, the result is expected. As the difference of the sensitivities between the synthetic and real crashes is not big, the results are quite reasonable. Third, for the undersampling data, only the SVM model could have better performance for the training data compared to the logistic regression model. However, for the test data, the sensitivity of the SVM model developed based on the

Table 7
Variable effects and model comparison.

Variables	Oversampling				Undersampling	
	DCGAN		SMOTE			
	Estimate	Std.	Estimate	Std.	Estimate	Std.
(Intercept)	−2.966**	0.057	−3.801**	0.0302	−0.475	1.577
volume_up1	–	–	–	–	1.005**	0.087
volume_down2	0.862**	0.002	0.847**	0.002	–	–
CI_down1	2.259**	0.020	2.812**	0.011	1.853**	0.564
avg_speed_up1	−1.265**	0.013	−1.038 **	0.007	−1.990**	0.368
speed_diff_up1	0.998**	0.001	0.950**	0.001	0.892**	0.0686
speeddiff_down1up1	0.003*	< 0.001	–	–	–	–
speeddiff_up1up2	0.010**	< 0.001	–	–	–	–

** Indicates 95% significance level.
* Indicates 90% significance level.

undersampling data is 0.628, which is much less than the sensitivity of the logistic model. The ANN and CNN models developed based on the undersampling data have significantly worse results compared to the corresponding logistic model. The result is expected since the neural network could not learn the trend with the small undersampling dataset. Finally, for the test data, the CNN model based on the DCGAN data could provide significantly better results compared to the counterparts. To be specific, the sensitivity and specificity of the CNN model based on the DCGAN data are 0.888 and 0.907, respectively, while the best sensitivity (the SVM model based on the SMOTE data) and specificity (the CNN model based on the SMOTE data) of the models based on the SMOTE and under-sampling data is 0.857 and 0.854, respectively. The results indicate that the CNN model based on the DCGAN data could have at least 3.6% and 6.2% higher sensitivity and specificity, compared to the models based on the SMOTE and under-sampling data. Based on the model fitting and prediction power of the training and test datasets, the DCGAN over-sampling method is preferred to balance the data for real-time crash predictions. The machine learning methods, especially the CNN model could better learn the big data and significantly improve the crash prediction accuracy for the over-sampling data (see Table 7).

5.3. Logistic modeling results

Table 3 summarizes the logistic modeling results with the significant variables. Three variables (i.e., CI_down1, avg_speed_up1, and speed_diff_up1) are significant in the three models. The signs of parameters for the three variables are consistent across the different models. The congestion index at the D1 location is found to increase the crash risk, and the average speed at the upstream is negatively associated with the crash occurrence. While congestion index is a direct indicator of congestion intensity, lower speed is also regarded as indirect measure of the congestion level. The higher speed difference between outer and inner lanes, indicating unstable traffic condition, would lead to higher crash risk. The volume at the D2 location is significant in the two over-sampling models while the volume at the U1 station is significant for the undersampling model. The volume variables are positively related to the crash risk. Two additional variables (i.e., speeddiff_down1up1 and speeddiff_up1up2) are only significant in the model based on the DCGAN over-sampling data. Both variables have positive effects on the crash occurrence, indicating that the crash risk tends to increase if the speed difference between upstream and downstream increases (if the speed at the upstream is higher than the speed at the downstream). The finding is consistent with the previous study (Xu et al., 2013a, 2013b). The result is not surprising since DCGAN could better capture the variable correlation by using the CNN structure while SMOTE only considered the value distance. In other words, the DCGAN could better capture the speed relation between the upstream and downstream locations.

Based on the result, variable speed limit should be used to gradually change the speed limits at the upstream to reduce the speed difference between upstream and downstream stations, if the crash probability has exceeded the predetermined thresholds. In addition, if the speed difference is over a predetermined range, alerts should be posted on the upstream dynamic message sign (DMS) to warning the upcoming drivers.

6. Conclusions

Real-time crash prediction is critical for proactive traffic safety management. In the past few decades, the traffic data has been grate enriched with the rapid development of traffic detection techniques. However, the traffic data for the crash and non-crash conditions are extremely imbalanced. Traditionally, most of the previous studies under sampled the data of non-crashes conditions to balance the data for developing real-time crash prediction models. However, such undersampling methods could not fully use the data of non-crash events, which may lose some important information due to the data heterogeneity. This study proposed a deep learning method called deep convolutional generative adversarial network (DCGAN) model to over sample the crash data. With the over-sampling crash data, the data for crash and non-crash cases could be balanced and the non-crash data could be fully used. To better capture the correlation between traffic variables, the data was augmented into a 2-D matrix. The suggested method was evaluated on expressways by comparing two other data balancing methods: (1) synthetic minority over-sampling technique

(SMOTE); and (2) random undersampling method. The data were divided into training and test datasets with the ratio of 7:3. The three data balance methods were applied for the training data. Four crash prediction algorithms including statistical logistic model, support vector machine (SVM), artificial neural network (ANN), and convolution neural network (CNN) were estimated based on the three balanced training data and in total twelve models were developed. The developed models were used to predict crashes in the test dataset for evaluating the model performance.

Among the two over-sampling methods, it was found that both the DCGAN and SMOTE methods could capture the data characteristics and distribution, while the DCGAN could provide a slightly smoother distribution. For the prediction performance, the statistical logistic regression models could achieve similar performance over the three balanced training data. The three machine learning algorithms could significantly improve the accuracy for the balanced data based on the two oversampling methods. However, for the undersampling data, the prediction accuracy of machine learning algorithms got worse compared to the logistic regression model. For the test data, the CNN crash prediction model based on the DCGAN-based balanced data has better performance considering both sensitivity and specificity. Meanwhile, the DCGAN- and SMOTE-based balanced data have similar performance in the SVM and ANN models. The results validated that the DCGAN model could be an alternative for crash data balance in addition to SMOTE, especially when the deep learning algorithms are applied for the crash risk prediction.

The three logistic regression models could identify significant variables including traffic volume, congestion index, speed difference between the outer and inner lanes, and average speed. In addition, the model based on DCGAN could identify additional important variables (i.e., speed difference between the upstream and downstream locations), which could guide the implementation of the traffic management strategies such as variable speed limits and dynamic message signs.

In summary, this study suggested a deep learning-based method to better balance data for developing real-time crash prediction models. With the suggested method, traffic safety could be further enhanced with more accurate prediction models and appropriate proactive traffic safety management strategies. However, there are still several limitations in the current study. For example, only a-year crash data on an expressway was used to develop the crash over-sampling model. To better capture the traffic characteristics associated to crashes, more data should be collected. Further, the procedures of data balancing and model estimation were separated to compare different data balancing methods across different models. As the best models for the two procedures are neural network models, it is expected that the prediction performance could be further improved if the two procedures are combined into a modeling structure. As suggested by Fiore et al. (2017), two discriminators could be included: one is to distinguish the synthetic examples produced by the generator from the real examples; and the second is to classify crashes and non-crashes.

CRediT authorship contribution statement

Qing Cai: Conceptualization, Methodology, Data curation, Investigation, Software, Writing - original draft. **Mohamed Abdel-Aty:** Supervision, Writing - review & editing. **Jinghui Yuan:** Data curation, Software. **Jaeyoung Lee:** Writing - review & editing. **Yina Wu:** Conceptualization, Investigation.

References

Abdel-Aty, M., Uddin, N., Pande, A., Abdalla, M.F., Hsia, L., 2004. Predicting freeway crashes from loop detector data by matched case-control logistic regression. Transp. Res. Rec.: J. Transp. Res. Board 1897 (1), 88–95.

Arjovsky, M., Bottou, L., 2017. Towards principled methods for training generative adversarial networks. arXiv 1701.04862.

Bao, J., Liu, P., Ukkusuri, S.V., 2019. A spatiotemporal deep learning approach for citywide short-term crash risk prediction with multi-source data. Accid. Anal. Prev. 122, 239–254.

Basso, F., Basso, L.J., Bravo, F., Pezoa, R., 2018. Real-time crash prediction in an urban expressway using disaggregated data. Transp. Res. Part C: Emerg. Technol. 86, 202–219.

Basso, F., Basso, L.J., Pezoa, R., 2020. The importance of flow composition in real-time crash prediction. Accid. Anal. Prev. 137, 05436.

Cai, Q., Abdel-Aty, M., Sun, Y., Lee, J., Yuan, J., 2019. Applying a deep learning approach for transportation safety planning by using high-resolution transportation and land use data. Transp. Res. Part A: Policy Pract. 127, 71–85.

Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. SMOTE: synthetic minority over-sampling technique. J. Artif. Interig. Res. 16, 321–357.

Elamrani Abou Elassad, Z., Mousannif, H., Al Moatassime, H., 2020. Class-imbalanced crash prediction based on real-time traffic and weather data: A driving simulator study. Traffic Inj. Prev. 21 (3), 201–208.

Fiore, U., De Santis, A., Perla, F., Zanetti, P., Palmieri, F., 2017. Using generative adversarial networks for improving classification effectiveness in credit card fraud detection. J. Inform. Sci. 479, 448–455.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets. Adv. Neural Inf. Process. Syst. 2672–2680.

He, H., Garcia, E.A., 2008. Learning from imbalanced data. IEEE Trans. Knowl. Data Eng. 9, 1263–1284.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S., 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. Adv. Neural Inf. Process. Syst. 6626–6637.

Huang, T., Wang, S., Sharma, A., 2020. Highway crash dectection and risk estimation using deep learning. Accid. Anal. Prev. 135, 105392.

Inkawhich, N., 2020. PyTorch DCGAN tutorial. https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html (accessed on 03/23/2020).

Ke, J., Zhang, S., Yang, H., Chen, X., 2019. PCA-based missing information imputation for real-time crash likelihood prediction under imbalanced data. Transportmet. A: Transp. Sci. 15 (2), 872–895.

Isola, P., Zhu, J.-Y., Zhou, T., Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1125–1134.

LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86 (11), 2278–2324.

Lee, C., Hellinga, B., Saccomanno, F., 2003. Real-time crash prediction model for application to crash prevention in freeway traffic. Transp. Res. Rec. 1840 (1), 67–77.

Li, Y., Li, Z., Wang, H., Wang, W., Xing, L., 2017a. Evaluating the safety impact of adaptive cruise control in traffic oscillations on freeways. Accid. Anal. Prev. 137–145.

Li, Y., Wang, H., Wang, W., Xing, L., Liu, S., Wei, X., 2017b. Evaluation of the impacts of cooperative adaptive cruise control on reducing rear-end collision risks on freeways. Accid. Anal. Prev. 98, 87–95.

Li, Y., Chen, Z., Yin, Y., Peeta, S., 2020a. Deployment of roadside units to overcome connectivity gap in transportation networks with mixed traffic. Transp. Res. Part C: Emerg. Technol. 111, 496–512.
Li, P., Abdel-Aty, M., Yuan, J., 2020b. Real-time crash risk prediction on arterials based on LSTM-CNN. Accid. Anal. Prev. 135, 105371.
Lin, L., Wang, Q., Sadek, A.W., 2015. A novel variable selection method based on frequent pattern tree for real-time traffic accident risk prediction. Transp. Res. Part C: Emerg. Technol. 55, 444–459.
Pande, A., Das, A., Abdel-Aty, M., Hassan, H., 2011. Estimation of real-time crash risk: are all freeways created equal? Transp. Res. Rec.: J. Transp. Res. Board 2237 (1), 60–66.
Roshandel, S., Zheng, Z., Washington, S., 2015. Impact of real-time traffic characteristics on freeway crash occurrence: Systematic review and meta-analysis. Accid. Anal. Prev. 79, 198–211.
Scikit learn, 2020. https://github.com/scikit-learn/scikit-learn/blob/ef5cb84a/sklearn/linear_model/stochastic_gradient.py#L616 (accessed on 05/27/2020).
Schlögl, M., Stütz, R., Laaha, G., Melcher, M., 2019. A comparison of statistical learning methods for deriving determining factors of accident occurrence from an imbalanced high resolution dataset. Accid. Anal. Prev. 127, 134–149.
Schlögl, M., 2020. A multivariate analysis of environmental effects on road accident occurrence using a balanced bagging approach. Accid. Anal. Prev. 136, 105398.
Shi, Q., Abdel-Aty, M., 2015. Big data applications in real-time traffic operation and safety monitoring and improvement on urban expressways. Transp. Res. Part C: Emerg. Technol. 58, 380–394.
Wang, L., Abdel-Aty, M., Lee, J., Shi, Q., 2019a. Analysis of real-time crash risk for expressway ramps using traffic, geometric, trip generation, and socio-demographic predictors. Accid. Anal. Prev. 122, 378–384.
Wang, L., Abdel-Aty, M., Ma, W., Hu, J., Zhong, H., 2019b. Quasi-vehicle-trajectory-based real-time safety analysis for expressways. Transp. Res. Part C: Emerg. Technol. 103, 30–38.
Wang, L., Zhong, H., Ma, W., Abdel-Aty, M., Park, J., 2020. How many crashes can connected vehicle and automated vehicle technologies prevent: A meta-analysis. Accid. Anal. Prev. 136, 105299.
Wang, L., Abdel-Aty, M., Shi, Q., Park, J., 2015. Real-time crash prediction for expressway weaving segments. Transp. Res. Part C: Emerg. Technol. 61, 1–10.
Wu, Y., Tan, H., Qin, L., Ran, B., Jiang, Z., 2018a. A hybrid deep learning based traffic flow prediction method and its understanding. Transp. Res. Part C: Emerg. Technol. 90, 166–180.
Wu, Y., Abdel-Aty, M., Lee, J., 2018b. Crash risk analysis during fog conditions using real-time traffic data. Accid. Anal. Prev. 114, 4–11.
Xu, C., Wang, W., Liu, P., 2013a. Identifying crash-prone traffic conditions under different weather on freeways. J. Saf. Res. 46, 135–144.
Xu, C., Tarko, A.P., Wang, W., Liu, P., 2013b. Predicting crash likelihood and severity on freeways with real-time loop detector data. Accid. Anal. Prev. 57, 30–39.
Yang, K., Wang, X., Yu, R., 2018. A Bayesian dynamic updating approach for urban expressway real-time crash risk evaluation. Transp. Res. Part C: Emerg. Technol. 96, 192–207.
Yu, R., Abdel-Aty, M., 2013. Utilizing support vector machine in real-time crash risk evaluation. Accid. Anal. Prev. 51, 252–259.
Yu, R., Abdel-Aty, M., 2014. An optimal variable speed limits system to ameliorate traffic safety risk. Transp. Res. Part C: Emerg. Technol. 46, 235–246.
Yu, R., Quddus, M., Wang, X., Yang, K., 2018. Impact of data aggregation approaches on the relationships between operating speed and traffic safety. Accid. Anal. Prev. 120, 304–310.
Yu, R., Wang, Y., Quddus, M., Li, J., Wang, X., Tian, Y., 2020. Investigating vehicle roadway usage patterns on the Shanghai urban expressway system and their impacts on traffic safety. Int. J. Sustain. Transp. 1–12.
Yuan, J., Abdel-Aty, M., Gong, Y., Cai, Q., 2019. Real-time crash risk prediction using long short-term memory recurrent neural network. Transp. Res. Rec.: J. Transp. Res. Board 0361198119840611.
Yuan, J., Abdel-Aty, M., Wang, L., Lee, J., Yu, R., Wang, X., 2018. Utilizing bluetooth and adaptive signal control data for real-time safety analysis on urban arterials. Transp. Res. Part C: Emerg. Technol. 97, 114–127.
Yuan, J., Abdel-Aty, M., Cai, Q., Yue, L., Gong, Y., 2020. Deep spatial-temporal network for corridor-level real-time crash risk prediction. Presented at the TRB 99th Annual Meeting.
Zhang, T., Damerau, F., Johnson, D., 2002. Text chunking based on a generalization of winnow. J. Mach. Learn. Res. 2, 615–637.
Zheng, Z., Ahn, S., Monsere, C.M., 2010. Impact of traffic oscillations on freeway crash occurrences. Accid. Anal. Prev. 42 (2), 626–636.
Zhu, X., Liu, Y., Qin, Z., Li, J., 2017. Data augmentation in emotion classification using generative adversarial networks. arXiv:.00648.