Accident Analysis and Prevention 135 (2020) 105392

Contents lists available at ScienceDirect

Accident Analysis and Prevention

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

# Highway crash detection and risk estimation using deep learning

Tingting Huang*, Shuo Wang, Anuj Sharma

*Department of Civil, Construction and Environmental Engineering, Iowa State University, United States*

ARTICLE INFO

*Keywords:*
Crash detection
Crash prediction
Deep learning

ABSTRACT

Crash Detection is essential in providing timely information to traffic management centers and the public to reduce its adverse effects. Prediction of crash risk is vital for avoiding secondary crashes and safeguarding highway traffic. For many years, researchers have explored several techniques for early and precise detection of crashes to aid in traffic incident management. With recent advancements in data collection techniques, abundant real-time traffic data is available for use. Big data infrastructure and machine learning algorithms can utilize this data to provide suitable solutions for the highway traffic safety system. This paper explores the feasibility of using deep learning models to detect crash occurrence and predict crash risk. Volume, Speed and Sensor Occupancy data collected from roadside radar sensors along Interstate 235 in Des Moines, IA is used for this study. This real-world traffic data is used to design feature set for the deep learning models for crash detection and crash risk prediction. The results show that a deep model has better crash detection performance and similar crash prediction performance than state of the art shallow models. Additionally, a sensitivity analysis was conducted for crash risk prediction using data 1-minute, 5-minutes and 10-minutes prior to crash occurrence. It was observed that is hard to predict the crash risk of a traffic condition, 10 min prior to a crash.

## 1. Introduction

According to the National Highway Traffic Safety Administration (NHTSA, 2017), there were 37,133 fatalities recorded in 2017 in the U.S. with 35 % of these crashes occurring on highways. Highway crashes can lead to enormous losses to our society in terms of personal injury, property damage and congestion on roadways. They also increase the risk of secondary crashes, especially on urban interstate highways. The U.S. Department of Transportation (DOT), state and local agencies are keen on curbing these road crash fatalities by identifying their causes and developing suitable prevention methods, working closely to meet the zero deaths vision.

To improve road safety and move towards zero fatality, data driven analysis plays an important role by providing insights from data. Generally, those analyses are addressing the safety problems in two different perspectives: planning and operation, with different emphases on explicability and accuracy. For planning perspective, it is important to analyze the causes and effects of crashes, with an emphasis on identifying and prioritizing the countermeasures. For example, many statistical models have been applied on analyzing crashes, with respect to road geometry, weather condition and traffic information. Based upon the results, one can plan to implement countermeasures to reduce crash or crash severity, such as installing cable median, rubble strips, etc. Another type of crash analysis is focusing on traffic operations and serve the needs of providing accurate risk information to road users, such as using machine learning method to detect incident or predict crash risk. This type of method focuses on improving the prediction accuracy, with less attentions on explaining the variables' effect. In fact, although explaining variables can be useful, some variables like weather condition are impossible to control. Thus, it is still vital that analyzing crash data at operation level because accurate and timely detection or prediction can potentially save time and lives.

With the emphasis on operation level analysis, this paper studies in detecting crashes and predicting crash risks using machine learning techniques. There have been many studies working on this area (Hossain et al., 2019). In machine learning domain, the accuracy is usually evaluated by classification measure such as area under precision recall curve or area under receiver operating characteristic (ROC) curve. Detailed methods and results will be elaborated in the following Literature Review section. Recently, deep learning achieves higher accuracy in some tasks (such as image classification, action recognition, etc.) due to its capability of complex feature learning, thus, researchers in transportation are also trying to improve the safety analysis by utilizing deep learning techniques. This paper will focus on designing deep neural networks and conducting experiments on parameter tuning, dataset choosing, to compare with traditional machine learning models.

* Corresponding author.
*E-mail address:* thuang1@iastate.edu (T. Huang).

https://doi.org/10.1016/j.aap.2019.105392
Received 8 April 2019; Received in revised form 2 December 2019; Accepted 2 December 2019
Available online 13 December 2019

With solving some issues in deep neural network training, this paper strives to provide some guide and best practice in applying deep learning on different type and amount of data.

Two types of experiments are conducted in this paper: crash detection and crash prediction, with high-resolution traffic flow information. We formulate the problem as a binary classification problem by using data during the event to train the detection classifier and using data before the event to train the prediction classifier. The model's prediction power is further tested using three types of historic data: a) 1 min before the crash; b) 5 min before the crash, and c) 10 min before the crash. Different deep neural network architectures have been tested and compared with common shallow classification models. Further, we leverage the training dataset size and model performance, to illustrate the data requirements in designing deep neural network architecture.

This study has two main contributions. First, this paper uses supervised learning approach to mine the pattern in large amount and high-resolution historical traffic data and detect unusual patterns caused by crashes, which can be implemented as an on-line tool for Traffic Incident Management (TIM) to help with early detection. Second, by investigating different prediction cases, one can select appropriate methods to estimate the crash risk ahead of crash occurrence, which provides additional information for safety monitoring.

## 2. Literature review

Traditional statistical methods aimed to analyze factors contributing to crash frequency on highways or intersections and predicted an aggregated number to represent the crash risk (Nassiri et al., 2014; Avelar et al., 2018). However, a classifer/predictor trained by machine learning techniques can provide more timely information as it focuses on each individual event. Common classification model like logistic regression, support vector machine (SVM), decision tree and neural network are widely used. Agarwal et al. (2016) developed a hybrid model using logistic regression with a wavelet-based feature extraction for detecting traffic incidents. Wang et al. (2019) built Bayesian logistic regression and SVM to predict the crash occurrence on ramp. Support Vector Machines (SVM) and its variants including multi-kernel SVM, AdaBoost SVM (Šingliar and Hauskrecht, 2007; Yuan and Cheu, 2003; Xiao and Liu, 2012; Chen et al., 2009; Wang et al., 2015) were popularly used for incident detection with various factors modeled like traffic condition, road geometry and weather. Other machine learning based methods include Decision trees (Chen and Wang, 2009), Back Propagation Neural Network (Cheng et al., 2010), Stochastic Gradient Boosting (Ahmed and Abdel-Aty, 2013), Nearest Neighbor model (Ozbayoglu et al., 2016) and Extreme Machine Learning (Li et al., 2017) were also applied on traffic incident detection. Some classification algorithms like Logistic Regression (Abdel-Aty et al., 2004; Xu et al., 2013; Wang et al., 2015a, b), Random Multinomial Logit (Hossain and Muromachi, 2013) and Probabilistic Neural Networks (Abdel-Aty and Pande, 2005; Oh et al., 2005) were also used for crash prediction, since the trained classifier can predict how similar a test case is with crash cases.

Although there are many algorithms used, the common idea behind them is learning the pattern from training data and identifying the case that has similar pattern with crash. This is also the main idea behind the method developed in this paper. Since these methods have a relatively small number of parameters used, thus, these methods are referred as shallow model in machine leaning domain.

Different from shallow models in machine learning, deep learning is a branch of machine learning that is structured with multi-layered neural networks. The number of layers can be expanded to even a hundred and the number of parameters can be thousands, thus it commonly is referred as deep model. Deep learning models are designed to learn representations from data without any human domain knowledge and have delivered state of the art accuracy in tasks such as object detection, speech recognition, language translation, and image analysis. They are popularly used with computer vision studies due to the rich data source available. In traffic engineering, some researchers have started adopting computer vision and deep learning techniques to solve traffic incident detection problems using video data (Wang et al., 2018; Zhu et al., 2018; El Hatri and Boumhidi, 2018). Similarly, researchers are also exploring the possibility of using deep learning in traffic incident prediction (Chen et al., 2016; Ren et al., 2017; Bao et al., 2019). This paper will use deep learning method on traffic condition to detect incident occurrence and estimate crash-prone traffic conditions. Instead of aggregating crashes into crash count in previous studies, this paper will consider individual crash case and its associated traffic condition, utilize the capability of deep learning to better capture the variability among all the cases and improve detection and prediction performance.

## 3. Data description and preprocessing

### 3.1. Crash data from traffic management centers reports

Traffic Management Centers (TMCs) of Iowa DOT are responsible for monitoring traffic signals, traffic flow and special events for Iowa's major streets and highway networks. Table 1 gives a sample data of TMC reports, which include incident ID, start and cleared timestamp, GPS coordinates of the incident location, corresponding road segment, and event type.

From the reports, only crash events are selected for this study. Further, to associate the crash with the correct traffic data, the GPS coordinates have been converted into mile markers of that specific roadway using Iowa DOT Roadway Asset Management System (RAMS). During the 2-year study period, there were 856 crashes reported on the target roadway segments.

### 3.2. Traffic data from radar sensors

Iowa DOT has installed hundreds of radar sensors along urban and rural freeways. These sensors collect real-time traffic information such as traffic volume (veh/h), average traffic speed (mi/h), and sensor time occupancy (%), at an interval of 20 s. Fig. 1 displays the locations of the sensors on the study area, Interstate 235 in Des Moines, IA. This freeway segment is a 13.78-mile long corridor that is frequently used in the Des Moines metropolitan area. There are 15 sensors on the eastbound direction and 14 sensors on the westbound direction with a mean spacing of 0.95 miles.

Data collected constitutes the traffic characteristics of the freeway segment such as traffic volume, speed and occupancy for every 20 s. Traffic volume is defined as the number of vehicles passing through a point on a highway during a specific time interval (20 s in this paper, same for below). Traffic speed is defined as the average speed of all

**Table 1**
A sample data of TMC crash reports.

| ID | Received | Cleared | Latitude | Longitude | Road | Direction | Event Type |
|---|---|---|---|---|---|---|---|
| 49644 | 2016-01-05 17:32:00 | 2016-01-05 18:53:00 | 41.592764 | −93.696594 | I235 | W | 3+ VEHICLE COLLISION |
| 49725 | 2016-01-08 07:18:00 | 2016-01-08 07:21:00 | 41.592766 | −93.736229 | I235 | E | 2 VEHICLE COLLISION |
| 50037 | 2016-01-13 08:03:00 | 2016-01-13 09:50:00 | 41.59268 | −93.694736 | I235 | E | 1 VEHICLE COLLISION |

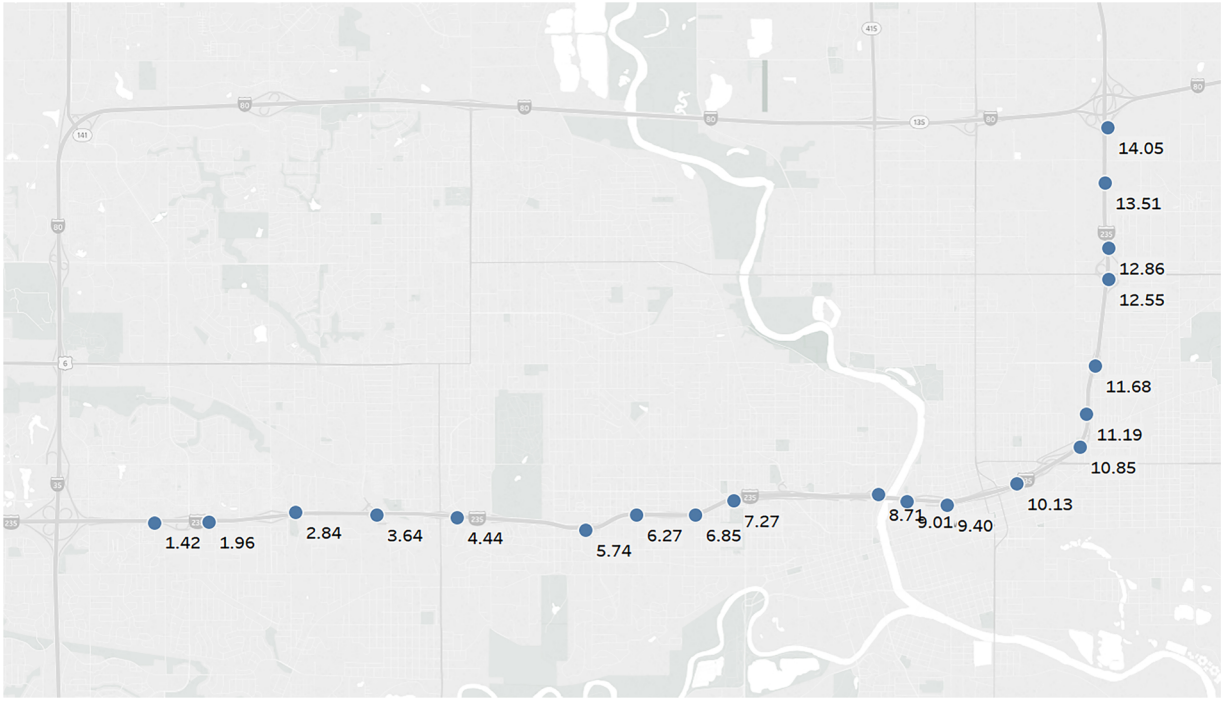



**Fig. 1.** Locations of sensors along Interstate 235 with mile marker labeled on westbound sensors.

vehicles passing through a point on a highway over a specified time period. Occupancy is the proportion of time that a detector is occupied by a vehicle in a defined time. To deal with the noise or missing information in such high-resolution data, this paper aggregates the 20 s interval raw data to 1-minute traffic volume, speed and occupancy data.

The 1-minute traffic data possesses some missing values due to short-term sensor malfunction. Thus, we fill the missing values by linear interpolation. Also, when no vehicle presents at night (volume is 0), the speed was set to 0 in raw data. To make sure the speed data represent the status of roadway, we also replace the zero speeds by interpolating it from before and after values. By doing this, we can have a continuous status change. Fig. 2(a) illustrates the heatmap of raw 1-minute speed data with time of day (x-axis) against sensor's mile marker (y-axis). Fig. 2(b) illustrates the temporal interpolation of speed data. For a complete day, missing values and zero speeds are replaced by linear interpolation of nearby speed values. Next, spatial interpolation is implemented based on the mile maker of sensors. Here we assume gradual change in terms of distance. The final spatial and temporal interpolated data are illustrated in Fig. 2(c).

### 3.3. Dataset generation

Fig. 3 illustrates how to generate sample data for crash detection and crash prediction. Crash detection focuses on the changes in traffic pattern after the occurrence of a crash. Positive samples (crash sample, labeled as 1 in classification problem) are generated from positive region. The blue shaded region in Fig. 3 shows the positive region, which starts 10 min before a crash is received by TMC to 10 min after the crash is cleared. In space, it covers the road from 2 miles upstream to 0.5 miles downstream. Every sample window has traffic data of 10 min in time and 1 mile in space ($10 \times 10 \times 3$ in data dimension). Multiple samples created by sliding the sample window 1 min in time and 0.1 mile in space within the positive region.

For Prediction we are focused on estimating the crash risk, hence, we sample data before crash occurrence. To identify the best historic data interval for estimating crash risk three different cases are studied. The positive samples are developed from data, 1 min before crash occurrence (case 1), 5 min before crash occurrence (case 2), and 10 min before crash occurrence (case 3) (see annotation in Fig. 3). For Prediction multiple positive samples are generated using only a sliding window in space. The space range is 0.5 mile in upstream and 2 miles in downstream to cope with the need of prediction for current location.

For both detection and prediction, the negative samples (non-crash samples, labeled as 0 in classification problem) are generated in areas where no crashes happened. (See white shaded area in Fig. 3 for illustration). To balance the positive samples and negative samples, the negative samples are generated randomly in any feasible region with the same dimension ($10 \times 10 \times 3$).

The total number of samples in each dataset are summarized in Table 2. Then, all the datasets are randomly split into training and testing dataset with ratio of 0.8 and 0.2. In addition, the day of week and time of day information is extracted for each sample and converted to weekend indicator and peak hour indicator (peak hour refers to 7–9 am and 4–6 pm inclusive in this paper).

## 4. Methodology

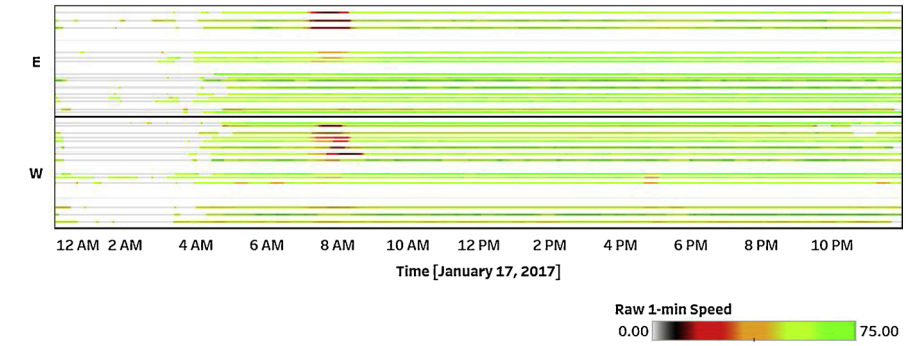
This study will use a Convolution Neural Network (CNN), a popular machine learning algorithm and a type of Artificial Neural Network (ANN). CNN was first introduced for image processing. It is widely used for studying the patterns in spatiotemporal traffic data and classifying the traffic conditions.

### 4.1. Artificial neural network

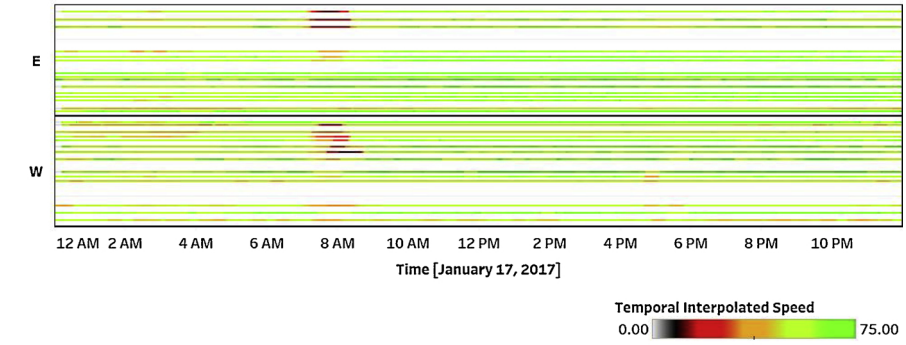
Typical ANN contains multiple neurons in different layers: an input layer, output layer and other hidden layers which cannot be completely explained. By tweaking the weights on each connection between neurons in different layers, the output approaches the truth, which is the labeled data (Cheng et al., 2010). This is the conventional supervised learning method. Several key techniques in ANN can be summarized as follows.

#### 4.1.1. Weight and Bias

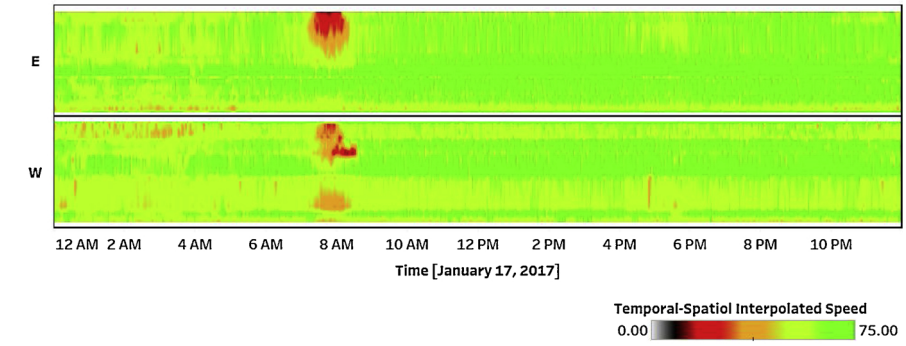
The connection between each layer are formed with learnable weights and bias. For example, one neuron in hidden layer $h$ can be




a) Raw 1-min Speed Heat Map




b) Temporal Interpolated Speed Heat Map




c) Spatial and Temporal Interpolated Speed Heat Map

**Fig. 2.** Traffic data interpolation (use speed data as an instance).

calculated by its input as:

$$h = w_1x_1 + w_2x_2 + ...+w_nx_n + b \quad (1)$$

Where, $w_1$, $w_2$, ..., $w_n$ are weights for input $x_1$, $x_2$, ..., $x_n$ respectively, and $b$ is the bias for neuron $h$.

Feeding forward the network, the output $y$ would be combining all the outputs from the previous hidden layers. Similar to Eq. 1, the output could also be computed by$y = w_1h_1 + w_2h_2 + ...+w_nh_n + b$. By comparing the ground truth (label) with the calculated output, the weights and biases are adjusted.

#### 4.1.2. Activation

To account for non-linearity, an activation function $f(x)$ is used to normalize the results. There are different activation functions available for use. Sigmoid function (Eq. 2) is a monotonic function that returns an output between 0 and 1. Besides sigmoid function, tanh function (Eq. 3), rectified linear unit (ReLU) (Eq. 4) and its alternative, exponential linear unit (ELU) (Eq. 5) are frequently used by researchers.

$$f(x) = sigmoid(x) = \frac{1}{1 + e^{-x}} \quad (2)$$

$$f(x) = \tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (3)$$

$$f(x) = \begin{cases} 0 for\ x < 0 \\ x\ for\ x \geq 0 \end{cases} \quad (4)$$

$$f(x) = \begin{cases} \alpha(e^{x} - 1)\ for\ x < 0 \\ x\ for\ x \geq 0 \end{cases} \quad (5)$$

In practice, the choice of activation function tends to be experimental. Different functions are tested in model training to select the function with the best performance. For the last layer of neural

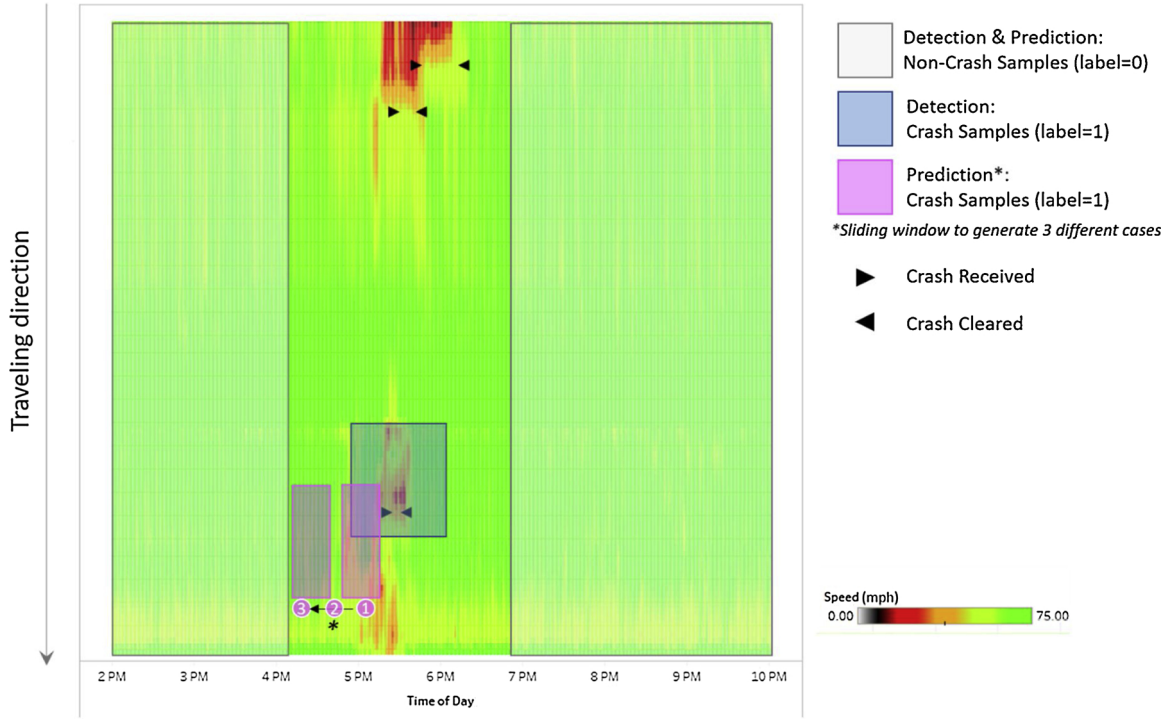



**Fig. 3.** Sample generation for different cases (1 detection case, 3 prediction cases).

**Table 2**
Number of samples in each dataset.

| Dataset | Train | | Test | | Total | |
|---|---|---|---|---|---|---|
| | Positive | Negative | Positive | Negative | Positive | Negative |
| Detection | 350,280 | 350,280 | 96,763 | 96,763 | 447,043 | 447,043 |
| Prediction Case 1 | 9,653 | 9,636 | 2,392 | 2,409 | 12,045 | 12,045 |
| Prediction Case 2 | 9,589 | 9,636 | 2,456 | 2,409 | 12,045 | 12,045 |
| Prediction Case 3 | 9,681 | 9,636 | 2,380 | 2,409 | 12,061 | 12,045 |

network, the output should match the labeled data. Therefore, for a multiclass problem, a multinomial logistic function (known as softmax function) is used. For $i^{th}$ class in total of $M$ classes, the probability of $i^{th}$ class is:

$$P(y=i)=\frac{e^{X^T W_i}}{\sum_{m=1}^{M} e^{X^T W_m}} \tag{6}$$

Where, $X$ is the input and $W$ is the weight.

#### 4.1.3. Loss function

The loss function is used to measure the difference between network output and the labeled data. With a particular object, the network can be trained to optimize it. For classification problem, cross entropy is preferred. The formula for cross entropy loss with $M > 2$ classes is as follows.

$$H=-\sum_{i=1}^{M} y^i \log \hat{y}^i \tag{7}$$

The loss function is also not deterministic for all model training. In this study, since we are dealing with a binary classification problem, the binary cross entropy loss function is used, which is $H=-(y log(\hat{y})+(1-y)\log(1-\hat{y}))$.

#### 4.1.4. Backpropagation

Neural network uses backpropagation method to update the weights iteratively to minimize the loss. The error will pass back to neurons and the weights are adjusted along the direction that reduces the loss. Consider a three-layer network: input layer ($X$), hidden layer ($h$) and output layer ($y$).

The error $E$ is calculated by selected loss function $E=L(y,\hat{y})$, where $y$ is the ground truth and $\hat{y}$ is the model prediction calculated by activation function $f_2$, $\hat{y}=f_2(h,\ W_2)$. $W_2$ is the weight from hidden layer to output layer, and $h$ is the activation from hidden layer calculated by activation function $f_1$, $h=f_1(X,\ W_1)$. $W_1$ is the weight from input layer to hidden layer. Thus, the partial derivative of $E$ on $W_1$ and $W_2$ are calculated by Eqs. 8 and 9 using chain rules.

$$\frac{\partial E}{\partial W_2}=\frac{\partial E}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial W_2} \tag{8}$$

$$\frac{\partial E}{\partial W_1}=\frac{\partial E}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial h}\frac{\partial h}{\partial W_1} \tag{9}$$

where,

$E$ is the error calculated using loss function $L$

$y$ is the ground truth and $\hat{y}$ is the model prediction calculated by activation function $f_2$

$h$ is the activation from hidden layer calculated by activation function $f_1$

$X$ is the input and $W_1$, $W_2$ are the weights on the input layer and hidden layer respectively.

To reduce the loss, the weights should be iteratively adjusted along the opposite direction of gradient, namely $\Delta W_i=-\alpha\frac{\partial E}{\partial W_i}$ with learning rate $\alpha$. This technique is called gradient descent.

Ideally, in each iteration the loss function should be reduced, and all the neurons should be adjusted based on their gradients. However, this is time consuming and requires high computation power when the training data is large. Therefore, one variant called stochastic gradient descent (SGD) is used to adjust the weights using few training data instead of the full training set.

### 4.2. Convolutional neural network

Convolutional neural network (CNN) was introduced for image processing and has been widely used in image classification problems.

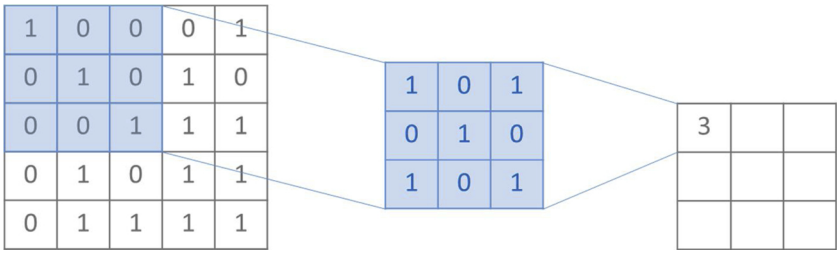



**Fig. 4.** Convolutional feature extraction.

**Table 3**
Overview of deep models on detection experiments.

| Model | Description* | Temporal Info | Regularization | Optimizer** | Number of Parameters |
|---|---|---|---|---|---|
| Model 1-1 | 3 conv and 1 fc | No | No | Adam | 15,336 |
| Model 1–2 | 3 conv and 1 fc | Yes | No | Adam | 15,338 |
| Model 1–3 | 3 conv and 1 fc | Yes | Yes | Adam | 15,338 |
| Model 2-1 | 4 conv and 1 fc | Yes | Yes | Adam | 29,674 |
| Model 3-1 | 3 conv and 2 fc | Yes | Yes | Adam | 24,042 |
| Model 4-1 | 2 conv and 2 fc | Yes | Yes | Adam | 40,426 |
| Model 5-1 | 1 conv and 2 fc | Yes | Yes | Adam | 34,026 |
| Model 5-2 | 1 conv and 2 fc | Yes | Yes | SGD | 34,026 |
| Model 6-1 | 1 conv and 1 fc | Yes | Yes | SGD | 8,426 |

* conv: convolutional layers; fc: fully connected layers;**Adam: adaptive moment estimation; SGD: stochastic gradient descent (with momentum in this paper).

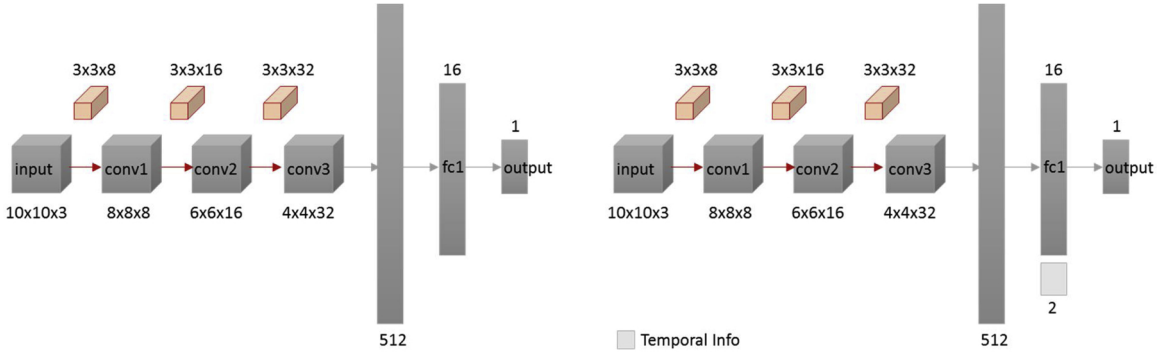



**Fig. 5.** Model 1 structure and variant.

**Table 4**
Deep model performance on crash detection.

| Model | Accuracy | F1-score | AUC-ROC | AUC-PR |
|---|---|---|---|---|
| model1-1 | 0.7459 | 0.7296 | 0.8173 | 0.8449 |
| model1–2 | 0.753 | 0.7352 | 0.8245 | 0.8520 |
| model1–3 | 0.7479 | 0.7389 | 0.8176 | 0.8496 |
| model2-1 | 0.7489 | 0.738 | 0.8182 | 0.8493 |
| model3-1 | 0.752 | 0.7437 | 0.8251 | 0.8539 |
| model4-1 | 0.7649 | 0.758 | 0.8389 | 0.8654 |
| model5-1 | 0.7734 | 0.7651 | 0.8463 | 0.8695 |
| model5-2 | 0.7685 | 0.7543 | 0.8398 | 0.8651 |
| model6-1 | 0.7451 | 0.7353 | 0.8223 | 0.8444 |

There are also some applications on traffic crash risk prediction (Ren et al., 2017; Bao et al., 2019). It has the ability of maintaining the local structure of an image and generating less parameters. CNN manages the spatial dependency through convolution. A small filter is applied that slides over the large image, to learn the features locally. The weights on the filter are shared with any locations of the image. Fig. 4 shows, a filter (3 by 3, in the middle) is applied on an input image (7 by 7, at the left) and the convolved feature (3 by 3, at the right) is calculated by element-wise multiplication and summation, $(1 \times 1 + 0 \times 0 + 0 \times 1 + 0 \times 0 + 1 \times 1 + 0 \times 0 + 0 \times 1 + 0 \times 0 + 1 \times 1) = 3$.

By weight sharing, a convolutional layer has less parameters than a fully connected layer. The local structure of an image is maintained, which makes it powerful in image classification. In this study, we treat the spatiotemporal traffic data as an image, and add time of day information as an additional node on fully connected layers, to classify if a specific traffic condition is crash-prone or not.

## 5. Experimental results

### 5.1. Crash detection

#### 5.1.1. Model structure and performance

Using an initial base model with multiple convolutional layers and fully connected layers, several different model structures have been tested and their performance is reported. Table 3 provides a short description and overview of the models tested. To better illustrate the model architecture, Fig. 5 shows the key components: number of layers, layer type, output dimension, kernel dimension and some special operations.

In training operation, data is scaled to 0–1 using min-max scaler.

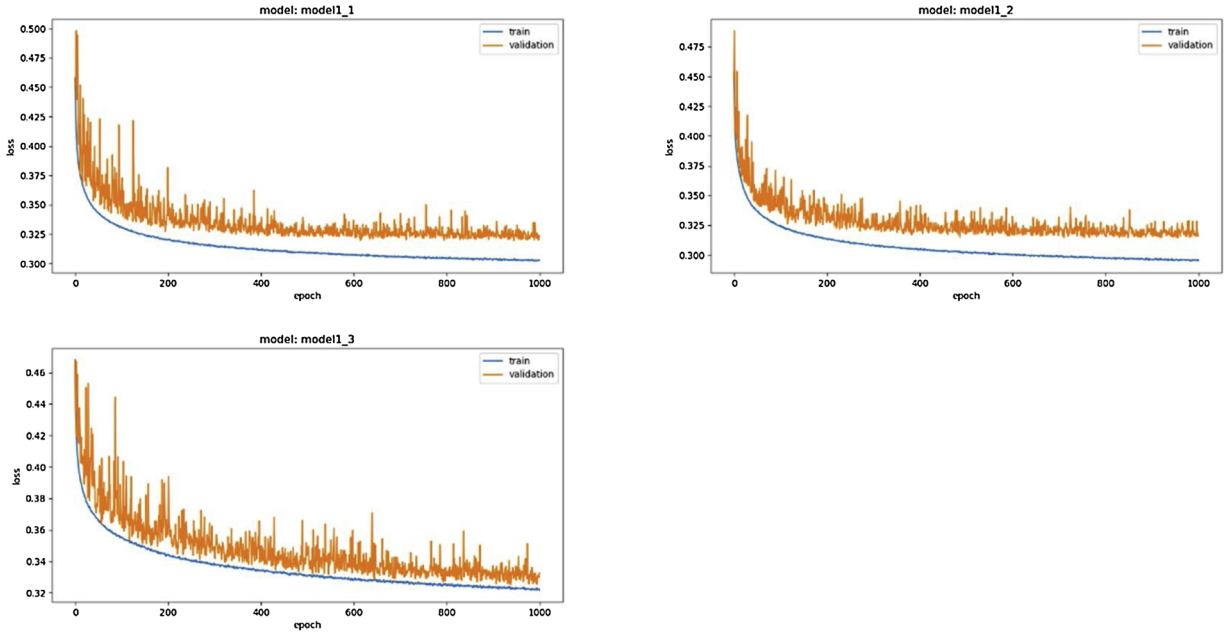



**Fig. 6.** Model 1 training loss.

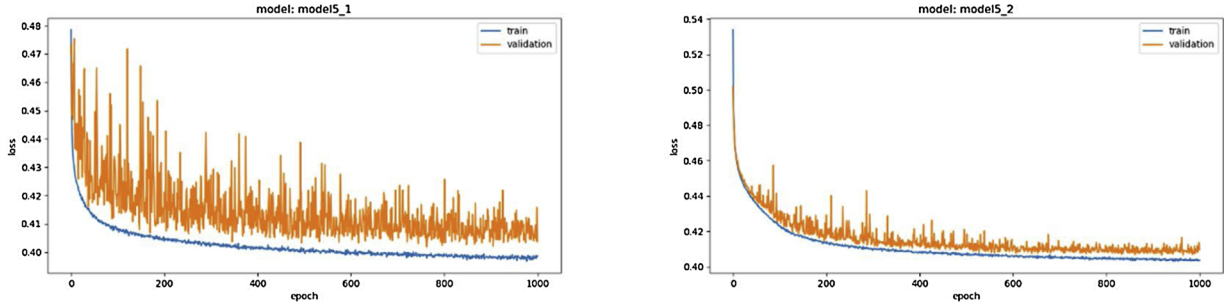



**Fig. 7.** Model 5 training loss.

**Table 5**
Shallow model performance on crash detection*.

| Model | Accuracy | F1-score | AUC-ROC | AUC-PR |
|---|---|---|---|---|
| LR-100 | 0.7297 | 0.6952 | 0.8249 | 0.8370 |
| LR-1000 | 0.7297 | 0.6952 | 0.8249 | 0.8370 |
| DT-0.05 | 0.7451 | 0.7197 | 0.8215 | 0.8401 |
| DT-0.005 | 0.7495 | 0.7285 | – | – |
| SVC-linear-C1 | 0.4296 | 0.6002 | – | – |
| SVC-linear-C10 | 0.5703 | 0.4579 | – | – |
| SVC-linear-C100 | 0.6094 | 0.4392 | – | – |
| SVC-rbf-C1 | 0.4616 | 0.6312 | – | – |
| SVC-rbf-C10 | 0.6429 | 0.5131 | – | – |
| SVC-rbf-C100 | 0.4229 | 0.5927 | – | – |
| SVC-rbf-C1-g0.1 | 0.3192 | 0.4289 | – | – |
| SVC-rbf-C10-g0.1 | 0.4105 | 0.4906 | – | – |
| SVC-rbf-C100-g0.1 | 0.3231 | 0.3682 | – | – |
| SVC-poly-3 | 0.5245 | 0.6318 | – | – |
| SVC-sigmoid | 0.6511 | 0.5411 | – | – |
| RF-90-0.005 | 0.7616 | 0.7431 | 0.8563 | 0.8678 |
| KNN-5 | 0.7149 | 0.6723 | 0.7625 | 0.8053 |

* Hyperparameters are indicated in model names, separated by underscores. LT: maximum iteration; DT: minimum samples percentage; SVC: kernel type, C value, and gamma value; RF: number of trees, minimum samples percentage; KNN: number of nearest neighbors.

Since mini-batch training technique is used, batch normalization is also applied in each layer. The regularization operation here is a drop-out operation with a rate of 0.5.

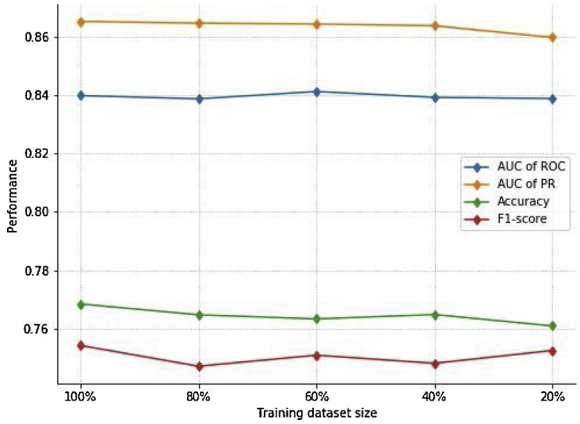
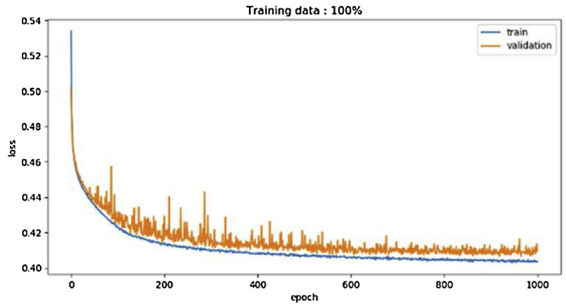


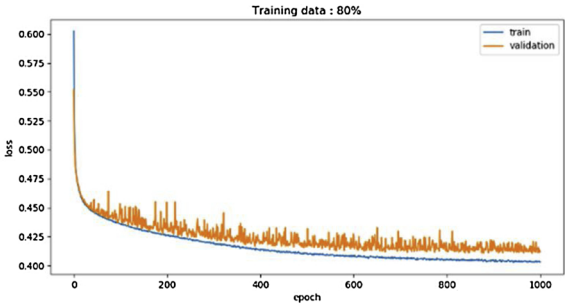

**Fig. 8.** Performance comparison with different data size.

Fig. 5(a) shows Model 1-1, a base model with 3 convolutional layers and 1 fully connected layer. To test the temporal information of data, two indicators are added. As described in 2.3, peak hour and weekend indicators are extracted and concatenated to the fully connected layer before output layer (Fig. 5b, Model 1–2).

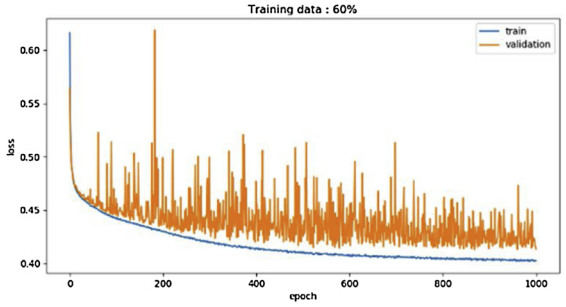
Model performances are reported and compared in Table 4. From the area under the curve (AUC) of receiver operating characteristic (ROC) and precision-recall (PR) curve, adding temporal information has a slightly better performance. However, when examining the training



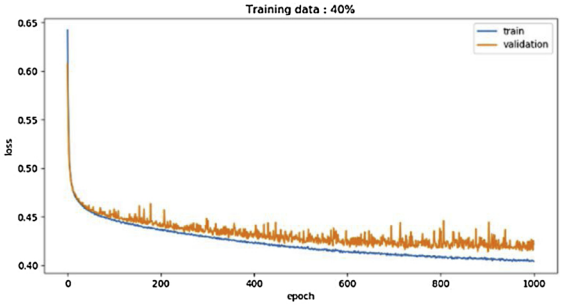

a) 100% training data



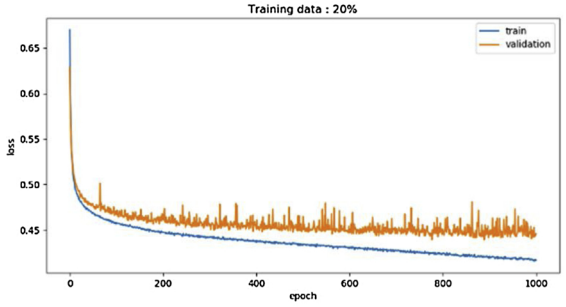

b) 80% training data




c) 60% training data




d) 40% training data




e) 20% training data

**Fig. 9.** Training loss comparison with different data size.

**Table 6**
Overview of deep models on prediction experiments.

| Model | Description* | GAP Operation | Optimizer** | Number of Parameters |
|---|---|---|---|---|
| Model 1-1 | 3 conv and 1 fc | Yes | Adam | 7,658 |
| Model 2-1 | 2 conv and 1 fc | Yes | Adam | 1,506 |
| Model 3-1 | 1 conv and 1 fc | Yes | Adam | 570 |
| Model 4-1 | 1 conv | Yes | Adam | 226 |
| Model 4-2 | 1 conv | Yes | SGD | 226 |
| Model 5-1 | 2 fc | No | Adam | 4,938 |
| Model 6-1 | 1 fc | No | Adam | 2,410 |
| Model 7-1 | 2 conv and 2 fc | No | Adam | 2,530 |

* conv: convolutional layers; fc: fully connected layers;**Adam: adaptive moment estimation; SGD: stochastic gradient descent (with momentum in this paper).

process (shown in Fig. 6), compared to Model 1–3, Model 1–1 and 1–2 show signs of overfitting. Thus, Model 1–3 which has regularization operation is preferred. Consequently, temporal information and dropout operation are used for all the following experiments.

To investigate the model structure, two approaches are used: making the model deeper (Model 2–1 and 3–1) or shallower (Model 4–1, 5–1, 5–2 and 6–1). According to Table 4, Model 5–1 has the best performance in terms of AUC of ROC and PR curve, however, by investigating the validation loss (Fig. 7), Model 5–1 tends to be fluctuated. Thus, the optimizer in training is changed from Adam (start learning rate of 0.001) to SGD (fixed learning rate of 0.001, momentum of 0.9), and re-trained 1000 epochs (iterations over the dataset). This could further reduce loss, because SGD with fixed learning rate may reduce the loss faster than Adam with adaptive rate (Ruder, 2016). Although the performance drops slightly from Model 5–1 to Model 5–2, a stable training process is preferred.

**Table 7**
Deep model performance on crash risk prediction.

a) AUC of ROC and PR curve

| Model | Case 1 | | Case 2 | | Case 3 | |
|---|---|---|---|---|---|---|
| | ROC | PR | ROC | PR | ROC | PR |
| model1-1 | 0.7699 | 0.7960 | 0.8288 | 0.8511 | 0.7594 | 0.7814 |
| model2-1 | 0.8674 | 0.8816 | 0.8353 | 0.8558 | 0.8116 | 0.8335 |
| model3-1 | 0.8744 | 0.8898 | 0.8781 | 0.8905 | 0.8425 | 0.8501 |
| model4-1 | 0.8722 | 0.8843 | 0.8881 | 0.8936 | 0.8570 | 0.8581 |
| model4-2 | 0.8705 | 0.8841 | 0.8900 | 0.8913 | 0.8606 | 0.8597 |
| model5-1 | 0.8572 | 0.8687 | 0.8829 | 0.8851 | 0.8272 | 0.8393 |
| model6-1 | 0.8386 | 0.8539 | 0.8552 | 0.8665 | 0.8414 | 0.8488 |
| model7-1 | 0.8004 | 0.8321 | 0.8362 | 0.8571 | 0.7845 | 0.8108 |

b) Accuracy and F1-score

| Model | Case 1 | | Case 2 | | Case 3 | |
|---|---|---|---|---|---|---|
| | Accuracy | F1- score | Accuracy | F1-score | Accuracy | F1-score |
| model1-1 | 0.7234 | 0.707 | 0.7737 | 0.7503 | 0.7043 | 0.6911 |
| model2-1 | 0.8013 | 0.7855 | 0.7743 | 0.7655 | 0.7212 | 0.7332 |
| model3-1 | 0.7986 | 0.7848 | 0.8025 | 0.7933 | 0.7734 | 0.7677 |
| model4-1 | 0.7911 | 0.7797 | 0.8029 | 0.791 | 0.7816 | 0.7686 |
| model4-2 | 0.7869 | 0.7748 | 0.8025 | 0.793 | 0.7757 | 0.7582 |
| model5-1 | 0.778 | 0.7544 | 0.7903 | 0.7788 | 0.7584 | 0.7463 |
| model6-1 | 0.754 | 0.7332 | 0.7694 | 0.7593 | 0.7609 | 0.7497 |
| model7-1 | 0.7609 | 0.7465 | 0.7519 | 0.7548 | 0.7133 | 0.7135 |

As in other machine learning methods, the same training and testing data are used, and several shallow models, such as logistic regression (LR), decision tree (DT), random forest (RF), support vector classification (SVC) and K-nearest neighbors (KNN) are studied. Among these shallow models, KNN is a non-parametric method where the function is only approximated locally, and all computation is deferred until classification; LR models the probability of a binary dependent variable; and SVM is a non-probabilistic binary classifier where the margin between training samples and the decision boundary is maximized. While DT is a tree-like model that only conditional statements are used as classification rules; and RF is an ensemble learning method that makes the classification by constructing multitude of DTs.

The hyperparameters in each model are also tuned and explained in the footnotes of Table 5. The best performance is accuracy of 0.76 and F1-score of 0.74 from random forest model.

Compared to shallow models, deep model has better performance in terms of accuracy and F-1 score for crash detection, however the margin is not large. One reason behind this observation can be that the samples are highly homogenous, because the crash data is sliding over the whole crash duration with 1-minute in time and 0.1 mile in space. This also explains why in deep models, not the largest model has the best performance. Model capacity should be carefully considered when applying deep learning. Another observation is the best shallow model (RF) even has slightly higher AUC of ROC and PR. Because random forest is an ensemble model with multiple trees, and the complexity can be much higher than other shallow models. In our case, we used 90 trees in random forest and achieved good performance, which indicates that when the model complexity increased, the performance from shallow model is also comparable with deep model.

#### 5.1.2. Model capacity and data size

Model capacity can be roughly defined as number of parameters in the model. From Table 3, the best deep model is not the largest one. To further investigate the relationship between deep model capacity and dataset size, an extra set of experiments have been conducted by applying the best deep model (Model 5–2) on different percentage of training data. Fig. 8 shows the relationship between model performance and percentage of total training samples. It can be seen that as the data size reduces, the model performance decreases slightly. In terms of the training process, Fig. 9 shows the training and validation loss over training steps for each reduced dataset. From this comparison, it is also clear that the model tends to overfit with less data. Thus, to apply deep learning methods, understanding the dataset and choosing an appropriate model structure are important.

### 5.2. Crash risk prediction

Crash risk prediction uses classification methods to provide prediction on test data. Prediction differs from detection in that the samples are extracted before the crash occurrence in time and space. As described in Section 2.3, three different prediction datasets are generated by using the data 1 min (case 1), 5 min (case 2) and 10 min (case 3) before crash. Since we only consider the 10-minute window of data before crash, the number of samples is much smaller for prediction compared to detection (which is sliding windows during the entire crash). Thus, instead of starting with a complex model such as for detection, prediction should consider relatively small models to accommodate the dataset size.

In deep learning, there are several methods that could reduce the number of parameters in a network. Global average pooling (GAP) has been applied instead of using fully connected layers after convolutional layers. GAP is pooling the layer along height and width dimension by computing the mean value and maintaining the channel dimension. This operation can reduce the number of learning parameters, and thereby help to avoid overfitting.

Table 6 highlights the models tested for the different prediction case studies. All the models have temporal information involved, drop-out operation with rate of 0.5 after each full connection, and batch normalization in each layer. Table 7 highlights the model performances. From Table 7, Model 4–2 using SGD is preferred which has a slightly lower performance than Model 4–1, but stable validation loss.

To compare with shallow models, some classification methods are tested. Different hyperparameter combination are also indicated in model name (see footnote for details). Their performance across 3 cases are listed in Table 8.

Overall, case 2 has the best performance and case 3 has the worst. Typically, the further away the data the harder it is to predict. However, prediction of crash occurrence may not be solely associated with traffic characteristics, other factors such as weather conditions or human behaviors could also contribute to a crash.

## 6. Conclusions

This paper applies deep learning algorithms including several model variants to solve traffic safety problems through crash detection and crash risk estimation on an urban Interstate highway. The detection and prediction problem are handled as a binary classification problem, with TMC crash reports with time and space information as labels, and the surrounding traffic data as samples. High-resolution traffic data including volume, speed and sensor occupancy are used in this study. After balancing the classes, different network structures and training operations have been explored. It can be found that for detection, convolutional neural networks (CNN) with drop-out operation performs better than some shallow models in terms of classification with a stable training process void of any overfitting issue. For prediction, data from different time slots are tested to further investigate the model prediction power. The results show that deep models have a comparable performance with shallow models, and it is worth noting that a less complex deep model can achieve better performance.

Overall, deep models can be applied to traffic data to classify crash occurrence or predict crash risk when appropriate labels are provided. However, careful inspection of model structure is needed, especially when data size is small. In this study, several concerns are also raised: a) data may have inaccurate labels b) TMC incident reports are generated

**Table 8**
Shallow model performance on crash risk prediction*.

a) AUC of ROC and PR curve

| Model* | Case 1 | | Case 2 | | Case 3 | |
|---|---|---|---|---|---|---|
| | ROC | PR | ROC | PR | ROC | PR |
| LR-100 | 0.8516 | 0.8657 | 0.8677 | 0.8795 | 0.8477 | 0.8473 |
| LR-1000 | 0.8516 | 0.8657 | 0.8677 | 0.8795 | 0.8477 | 0.8473 |
| DT-0.05 | 0.8448 | 0.8600 | 0.8694 | 0.8790 | 0.8208 | 0.8070 |
| DT-0.005 | 0.8096 | 0.8295 | 0.8338 | 0.8680 | 0.7681 | 0.7754 |
| RF-0.005(170/190/130) | 0.8664 | 0.8806 | 0.9098 | 0.9189 | 0.8584 | 0.8588 |
| KNN-5 | 0.8081 | 0.8403 | 0.8618 | 0.8929 | 0.7923 | 0.8148 |

b) Accuracy and F1-score

| Model* | Case 1 | | Case 2 | | Case 3 | |
|---|---|---|---|---|---|---|
| | Accuracy | F1-score | Accuracy | F1-score | Accuracy | F1-score |
| LR-100 | 0.7754 | 0.7572 | 0.7870 | 0.7759 | 0.7604 | 0.7451 |
| LR-1000 | 0.7754 | 0.7572 | 0.7870 | 0.7759 | 0.7604 | 0.7451 |
| DT-0.05 | 0.7602 | 0.7440 | 0.7730 | 0.7587 | 0.7531 | 0.7322 |
| DT-0.005 | 0.7540 | 0.7334 | 0.7671 | 0.7381 | 0.6859 | 0.6552 |
| SVC-linear-C1 | 0.7702 | 0.7488 | 0.7901 | 0.7776 | 0.7646 | 0.7454 |
| SVC-linear-C10 | 0.7706 | 0.7494 | 0.7909 | 0.7791 | 0.7617 | 0.7417 |
| SVC-linear-C100 | 0.7717 | 0.7506 | 0.7895 | 0.7773 | 0.7644 | 0.7454 |
| SVC-rbf-C1 | 0.7827 | 0.7683 | 0.7901 | 0.7823 | 0.7669 | 0.7511 |
| SVC-rbf-C10 | 0.7908 | 0.7779 | 0.7991 | 0.7918 | 0.7744 | 0.7626 |
| SVC-rbf-C100 | 0.7935 | 0.7815 | 0.8037 | 0.7965 | 0.7780 | 0.7658 |
| SVC-rbf-C1-g0.1 | 0.7933 | 0.7766 | 0.7942 | 0.7816 | 0.7813 | 0.7631 |
| SVC-rbf-C10-g0.1 | 0.8042 | 0.7887 | 0.7958 | 0.7779 | 0.7707 | 0.7460 |
| SVC-rbf-C100-g0.1 | 0.7735 | 0.7447 | 0.7726 | 0.7365 | 0.7183 | 0.6614 |
| SVC-rbf-C1-g0.01 | 0.7915 | 0.7791 | 0.7960 | 0.7892 | 0.7726 | 0.7586 |
| SVC-rbf-C10-g0.01 | 0.7952 | 0.7831 | 0.8014 | 0.7942 | 0.7797 | 0.7669 |
| SVC-rbf-C100-g0.01 | 0.7927 | 0.7784 | 0.7965 | 0.7855 | 0.7755 | 0.7577 |
| RF-0.005(170/190/130) | 0.7950 | 0.7746 | 0.8221 | 0.8067 | 0.7746 | 0.7481 |
| KNN-5 | 0.7640 | 0.7468 | 0.7979 | 0.7856 | 0.7366 | 0.7134 |

* Hyperparameters are indicated in model names, separated by underscores. LT: maximum iteration; DT: minimum samples percentage; SVC: kernel type, C value, gamma value; RF: minimum samples percentage, number of trees; KNN: number of nearest neighbors.

when a crash is reported, thus, they may have temporal and spatial differences relative to the true crash; c) data is relatively homogenous. When generating crash samples, the target window is sliding with 1 min in time and 0.1 mile in space. Hence, lots of crash samples are created from one crash and have very similar patterns. This is a common issue in crash studies because of the sparseness of crash data; For prediction, traffic characteristics are related to crash risk, but they are not necessarily the only contributors. Other factors as adverse weather condition, driver distraction, etc. can also lead to a crash. This paper has the limitation of data explored, which works on the incidents with traffic impacts, and could be improved when considering more variable inputs.

Traffic safety is the key concern in traffic engineering and management. Research efforts are being dedicated to diagnosing the traffic safety system. In recent times as computation power increases, the feasibility of applying deep learning in varied research areas have become a reality. Deep learning serves as an efficient approach to researchers to solve generic, complex problems. Similarly, this study demonstrates the ability to utilize such deep learning algorithms to detect crash and estimate crash risk. Considering the efficiency of this method, more comprehensive data, and study locations should be included to test the performance and generalization of deep models in the future.

## CRediT authorship contribution statement

**Tingting Huang:** Methodology, Software, Formal analysis, Investigation, Data curation, Writing - original draft, Visualization. **Shuo Wang:** Validation, Investigation, Data curation, Software. **Anuj Sharma:** Conceptualization, Resources, Writing - review & editing, Supervision.

## Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

Abdel-Aty, M., Pande, A., 2005. Identifying crash propensity using specific traffic speed conditions. J. Safety Res. 36 (1), 97–108. https://doi.org/10.1016/j.jsr.2004.11.002.
Abdel-Aty, M., Uddin, N., Pande, A., Abdalla, 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, 88–95. https://doi.org/10.3141/1897-12.
Agarwal, S., Kachroo, P., Regentova, E., 2016. A hybrid model using logistic regression and wavelet transformation to detect traffic incidents. IATSS Res. 40 (1), 56–63. https://doi.org/10.1016/j.iatssr.2016.06.001.
Ahmed, M., Abdel-Aty, M., 2013. A data fusion framework for real-time risk assessment on freeways. Transp. Res. Part C Emerg. Technol. 26, 203–213. https://doi.org/10.1016/j.trc.2012.09.002.
Avelar, R.E., Dixon, K., Ashraf, S., 2018. A comparative analysis on performance of severe crash prediction methods. Transp. Res. Rec.: J. Transp. Res. Board Retrieved from. https://trid.trb.org/view/1494454.
Bao, J., Liu, P., Ukkusuri, S.V., 2019. A spatiotemporal deep learning approach for city-wide short-term crash risk prediction with multi-source data. Accid. Anal. Prev. 122, 239–254. https://doi.org/10.1016/j.aap.2018.10.015.
Chen, Q., Song, X., Yamada, H., Shibasaki, R., 2016. Learning deep representation from Big and heterogeneous data for traffic accident inference. AAAI 338–344.
Chen, S., Wang, W., 2009. Decision tree learning for freeway automatic incident detection. Expert Syst. Appl. 36 (2), 4101–4105. https://doi.org/10.1016/j.eswa.2008.03.012.
Chen, S., Wang, W., van Zuylen, H., 2009. Construct support vector machine ensemble to detect traffic incident. Expert Syst. Appl. 36 (8), 10976–10986. https://doi.org/10.

1016/j.eswa.2009.02.039.
Cheng, X., Lin, W., Liu, E., Gu, D., 2010. Highway traffic incident detection based on BPNN. Procedia Eng. 7, 482–489. https://doi.org/10.1016/j.proeng.2010.11.080.
El Hatri, C., Boumhidi, J., 2018. Fuzzy deep learning based urban traffic incident detection. Cogn. Syst. Res. 50, 206–213. https://doi.org/10.1016/j.cogsys.2017.12.002.
Hossain, M., Abdel-Aty, M., Quddus, M.A., Muromachi, Y., Sadeek, S.N., 2019. Real-time crash prediction models: state-of-the-art, design pathways and ubiquitous requirements. Accid. Anal. Prev. 124, 66–84. https://doi.org/10.1016/j.aap.2018.12.022.
Hossain, M., Muromachi, Y., 2013. Understanding crash mechanism on urban expressways using high-resolution traffic data. Accid. Anal. Prev. 57, 17–29. https://doi.org/10.1016/j.aap.2013.03.024.
Li, L., Qu, X., Zhang, J., Ran, B., 2017. Traffic incident detection based on extreme machine learning. J. Appl. Sci. Eng. 20 (4), 409–416. https://doi.org/10.6180/jase.2017.20.4.01.
Nassiri, H., Najaf, P., Amiri, A.M., 2014. Prediction of roadway accident frequencies: count regressions versus machine learning models. Sci. Iranica. Trans. A, Civil Eng. 21 (2), 263.
National Highway Traffic Safety Administration, 2017. Fatal Motor Vehicle Crashes: Overview. Accessed at. https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812603.
Oh, C., Oh, J.S., Ritchie, S.G., 2005. Real-time hazardous traffic condition warning system: framework and evaluation. IEEE Trans. Intell. Transp. Syst. 6 (3), 265–272. https://doi.org/10.1109/TITS.2005.853693.
Ozbayoglu, A.M., Kucukayan, G., Dogdu, E., 2016. A Real-time Autonomous highway accident detection model based on big data processing and computational intelligence. 2016 IEEE International Conference on Big Data (Big Data) 1807–1813. https://doi.org/10.1109/BigData.2016.7840798.
Ren, H., Song, Y., Wang, J., Hu, Y., Lei, J., 2017. A Deep Learning Approach to the Citywide Traffic Accident Risk Prediction. ArXiv:1710.09543 [Cs]. Retrieved from. http://arxiv.org/abs/1710.09543.
Ruder, S., 2016. An Overview of Gradient Descent Optimization Algorithms. arXiv preprint arXiv:1609.04747. .
Šingliar, T., Hauskrecht, M., 2007. Learning to detect adverse traffic events from noisily labeled data. Knowledge Discovery in Databases: KDD 2007. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 236–247. https://doi.org/10.1007/978-3-540-74976-9. 4702.
Wang, L., Abdel-Aty, M., Shi, Q., Park, J., 2015a. Real-time crash prediction for expressway weaving segments. Transp. Res. Part C Emerg. Technol. 61, 1–10. https://doi.org/10.1016/j.trc.2015.10.008.
Wang, L.-L., Ngan, H.Y.T., Yung, N.H.C., 2015. Automatic Incident Classification for Big Traffic Data by Adaptive Boosting SVM. ArXiv:1512.04392 [Cs]. Retrieved from. http://arxiv.org/abs/1512.04392.
Wang, L., Shi, Q., Abdel-Aty, M., 2015b. Predicting crashes on expressway ramps with real-time traffic and weather data. Transp. Res. Rec.: J. Transp. Res. Board 2514, 32–38. https://doi.org/10.3141/2514-04.
Wang, B., Zhu, Y., Shen, Y., Liu, Q., 2018. Traffic conflict detection of vehicle and Non-motorized vehicle at intersection based on deep learning. Presented at the Transportation Research Board 97th Annual Meeting Transportation Research Board Retrieved from. https://trid.trb.org/view/1494523.
Wang, L., Abdel-Aty, M., Lee, J., Shi, Q., 2019. Analysis of real-time crash risk for expressway ramps using traffic, geometric, trip generation, and socio-demographic predictors. Accid. Anal. Prev. 122, 378–384. https://doi.org/10.1016/j.aap.2017.06.003.
Xiao, J., Liu, Y., 2012. Traffic incident detection using multiple-kernel support vector machine. Transp. Res. Rec.: J. Transp. Res. Board 2324 (1), 44–52. https://doi.org/10.3141/2324-06.
Xu, C., Tarko, A.P., Wang, W., Liu, P., 2013. Predicting crash likelihood and severity on freeways with real-time loop detector data. Accid. Anal. Prev. 57, 30–39. https://doi.org/10.1016/j.aap.2013.03.035.
Yuan, F., Cheu, R.L., 2003. Incident detection using support vector machines. Transp. Res. Part C Emerg. Technol. 11 (3–4), 309–328. https://doi.org/10.1016/S0968-090X(03)00020-2.
Zhu, L., Guo, F., Krishnan, R., Polak, J.W., 2018. The use of convolutional neural networks for traffic incident detection at a network level. Presented at the Transportation Research Board 97th Annual Meeting Transportation Research Board Retrieved from. https://trid.trb.org/view/1494378.