applied sciences

MDPI

Article

Severity Prediction of Traffic Accidents with Recurrent Neural Networks

Maher Ibrahim Sameen † and Biswajeet Pradhan †,*

Department of Civil Engineering, Geospatial Information Science Research Center (GISRC), Faculty of Engineering, University Putra Malaysia, UPM, Serdang 43400, Malaysia; maherrsgis@gmail.com

* Correspondence: biswajeet24@gmail.com or biswajeet@upm.edu.my; Tel.: +60-3-8946-6383

† These authors contributed equally to this work.

Academic Editor: Saro Lee

Received: 14 March 2017; Accepted: 28 April 2017; Published: 8 June 2017

Abstract: In this paper, a deep learning model using a Recurrent Neural Network (RNN) was developed and employed to predict the injury severity of traffic accidents based on 1130 accident records that have occurred on the North-South Expressway (NSE), Malaysia over a six-year period from 2009 to 2015. Compared to traditional Neural Networks (NNs), the RNN method is more effective for sequential data, and is expected to capture temporal correlations among the traffic accident records. Several network architectures and configurations were tested through a systematic grid search to determine an optimal network for predicting the injury severity of traffic accidents. The selected network architecture comprised of a Long-Short Term Memory (LSTM) layer, two fully-connected (dense) layers and a Softmax layer. Next, to avoid over-fitting, the dropout technique with a probability of 0.3 was applied. Further, the network was trained with a Stochastic Gradient Descent (SGD) algorithm (learning rate = 0.01) in the Tensorflow framework. A sensitivity analysis of the RNN model was further conducted to determine these factors' impact on injury severity outcomes. Also, the proposed RNN model was compared with Multilayer Perceptron (MLP) and Bayesian Logistic Regression (BLR) models to understand its advantages and limitations. The results of the comparative analyses showed that the RNN model outperformed the MLP and BLR models. The validation accuracy of the RNN model was 71.77%, whereas the MLP and BLR models achieved 65.48% and 58.30% respectively. The findings of this study indicate that the RNN model, in deep learning frameworks, can be a promising tool for predicting the injury severity of traffic accidents.

Keywords: severity prediction; GIS; traffic accidents; deep learning; recurrent neural networks

1. Introduction

Traffic accidents are a primary concern due to many fatalities and economic losses every year worldwide. In Malaysia, recent statistics show that there are nearly 24 deaths per 100,000 people for all road users [1]. Expressways are potential sites of fatal highway accidents in Malaysia. Better accident severity prediction models are critical to enhancing the safety performance of road traffic systems. According to recent literature, driver injury severity can be classified into a few categories such as property damage, possible/evident injury, or disabling injury/fatality [2]. Therefore, modeling accident severity can be addressed as a pattern recognition problem [3], which can be solved by deep learning, statistical techniques and sometimes by physical modelling approaches [4–7]. In a deep learning model, an input vector is often mapped into an output vector through a set of nonlinear functions. In the case of accident severity, the input vectors are the characteristics of the accident, such as driver behavior and highway, vehicle and environment characteristics. The output vector is the corresponding classes of accident severity. Deep learning allows computational models to learn hierarchal representations of data with multiple levels of abstraction at different processing layers. The advantage of deep learning neural networks over statistical techniques is that they involve a more general mapping procedure i.e., a specific function is not required in model building [8]. However, these techniques can be treated as black box methods, if the network architecture is not carefully designed and its parameters are not optimized.

Several studies have investigated Neural Network (NN) and deep learning methods in transportation related applications [9–12]. For example, Abdelwahab and Abdel-Aty [8] used NN to predict driver injury severity from various accident factors (i.e., driver, vehicle, roadway and environment characteristics). In their study, the NN model developed performed better than the ordered probit model. In another paper, Delen et al. [10,13] applied NN to model injury severity of road accidents using 17 significant parameters. The NN model was used to predict the injury severity levels, resulting in a low overall accuracy of 40.71%. More recently, Hashmienejad and Hasheminejad [14] proposed a novel rule-based technique to predict traffic accident severity based on users' preferences instead of conventional data mining methods. The proposed method outperformed the NN and support vector machine methods. In a recent paper, Alkheder et al. [11] used NN methods to predict the injury severity (minor, moderate, severe, death) of traffic accidents in Abu Dhabi. Their analysis was based on 5973 traffic accident records that had occurred over a 6-years period. The overall accuracy of the model for the training and testing data were 81.6% and 74.6%, respectively. In addition, Zeng and Huang [15] proposed a training algorithm and network structure optimization method for crash injury severity prediction. Their results indicated that the proposed training algorithm performed better than the traditional back-propagation algorithm. The optimized NN, which contained less nodes than the fully connected NN, achieved reasonable prediction accuracy. Also, the fully connected and optimized NN models outperformed the ordered logit model. Also, their results showed that optimization of the NN structure could improve the overall performance of model prediction.

In recent years, deep learning has become a popular technique in image [16] and natural language processing applications [17]. Many researchers have applied deep learning based techniques in transportation related applications such as traffic flow [18] and accident hotspots prediction [19]. A detailed literature review showed that optimization of network structures is critical for crash severity prediction. Generally speaking, deep learning allows compositionality and the design of flexible network structures with multilayer modules, and therefore it is expected that the performance of deep learning models will be better than the traditional NN models. A deep learning architecture is a multilayer stack of simple modules, most of which are subjected to learning computing nonlinear input–output mappings [20]. Each module in the stack transforms its input to increase both selectivity and invariance of the representation. A Recurrent Neural Network (RNN) is a type of deep learning module, which is more appropriate for sequential data (e.g., traffic accident data with temporal correlations). It is effective for a wide range of applications including text generation [21] and speech recognition [22]. The application of RNNs in various fields motivated the authors to evaluate these models in deep learning frameworks for severity prediction of traffic accidents.

The main novelty of this work is the development of a RNN model that accurately predicts injury severity of traffic accidents utilizing the temporal structure of accident data. The specific objective is to design a deep learning model using a RNN for severity prediction of traffic accidents. The hyperparameters of the model will be selected through a systematic grid search technique. In addition, the sensitivity of model parameters and configurations will be assessed, and then will be compared with the well-known Multilayer Perceptron (MLP) and Bayesian Logistic Regression (BLR) models to understand its advatange. Finally, the impacts of accident-related factors on injury severity outcome will be determined using the profile-based method.

2. Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are neural networks with feedback connections specifically designed to model sequences. They are computationally more powerful and biologically more reasonable than feed-forward networks (no internal states). The feedback connections provide a RNN the memory of past activations, which allows it to learn the temporal dynamics of sequential data. A RNN is powerful because it uses contextual information when mapping between input and output sequences. However, the traditional RNNs have a problem called vanishing or exploding gradient. To handle this problem, Hochreiter and Schmidhuber [23] proposed the Long Short-Term Memory (LSTM) algorithm.

In LSTM, the hidden units are replaced by memory blocks, which contain one or more self-connected memory cells and three multiplicative units (input, output, forget gates). These gates allow writing, reading, and resetting operations within a memory block, and they control the overall behavior internally. A representation of a single LSTM unit is shown in Figure 1. Let c_t be the sum of inputs at time step t, then LSTM updates for time step i at given inputs x_t, h_{t-1}, and c_{t-1} are [24]:

$$i_t = \sigma(W_{xi}.x_t + W_{hi}.h_{t-1} + W_{ci}.c_{t-1} + b_i) \tag{1}$$

$$f_t = \sigma\left(W_{xf}.x_t + W_{hf}.h_{t-1} + W_{cf}.c_{t-1} + b_f\right) \tag{2}$$

$$c_t = i_t.tanh(W_{xc}.x_t + W_{hc}.h_{t-1} + b_c) + f_t.c_{t-1} \tag{3}$$

$$o_t = \sigma(W_{xo}.x_t + W_{ho}.h_{t-1} + W_{co}.c_t + b_o) \tag{4}$$

$$h_t = o_t.\tanh(c_t) \tag{5}$$

where σ is an element-wised non-linearity such as a sigmoid function, W is the weight matrix, x_t is the input at time step t, h_{t-1} is the hidden state vector of the previous time step and b_i denotes the input bias vector.

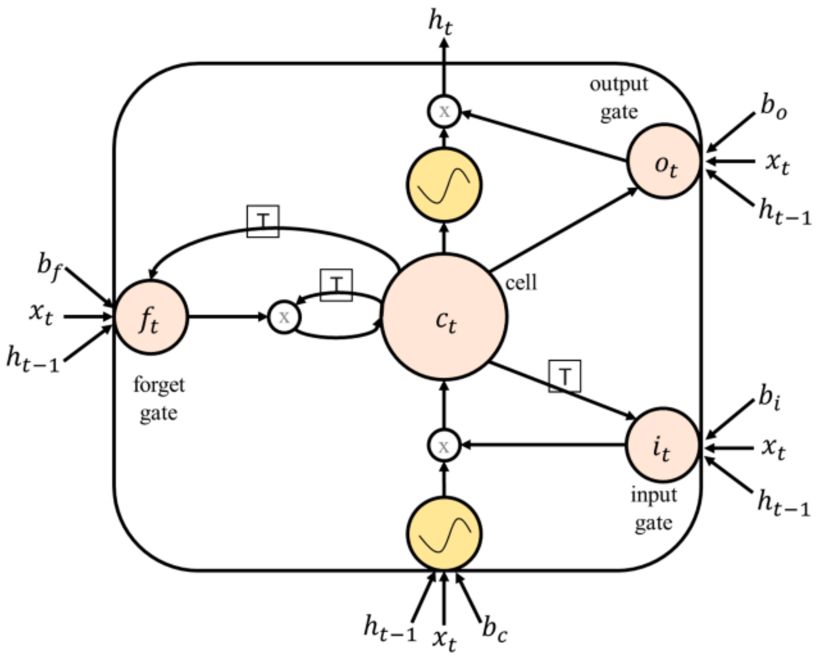

Figure 1. The structure of a memory cell in the Long Short-Term Memory–Recurrent Neural Network (LSTM–RNN).

3. The Proposed Network Framework

3.1. Network Architecture

Figure 2 shows the high-level architecture of the proposed severity prediction model based on deep learning. The advantage of LSTM-RNN over the traditional neural networks is that the traffic accidents are correlated to the historical and future incidents [25]. The RNN model comprises of five main layers which include input, LSTM, two dense layers and a Softmax layer.

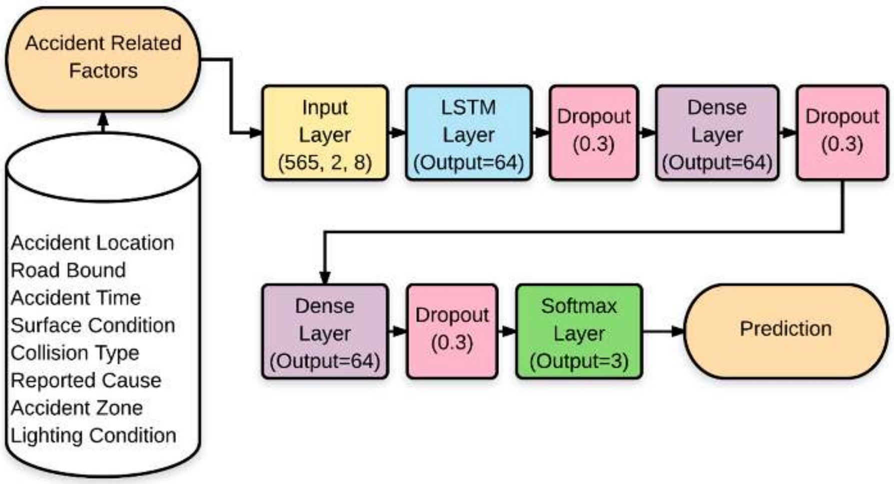

Figure 2. The high-level architecture of the proposed RNN model used in this study.

The main input of the network is a set of traffic accident-related factors, and the output is the corresponding accident severity classes (i.e., property damage only, possible/evident injury, disabling/fatality). The input dimension of the LSTM layer is equal to eight (number of input factors) with 64 nodes. The output of each node in the LSTM layer is results from the Rectified Linear Unit (ReLU) activation function applied to a weighted sum of both inputs from the previous layer and the previous outputs of the layer. The ReLU computes the function $f(x) = \max(0, x)$. This activation function accelerates (e.g., a factor of 6 in [26]) the convergence of Stochastic Gradient Descent (SGD) compared to the sigmoid/tanh functions. In addition, it can be implemented without expensive operations as in the case of sigmoid/tanh functions. Then, two fully connected layers were trained on top of the LSTM layers. They were added to match the output of the LSTM layer and the size of the accident severity classes. The output layer was a fully connected feed-forward layer and used to directly map the learned features to the three accident severity classes. A Softmax function is used to activate the output layer. In addition, three dropout layers with probability of 0.3 were used to reduce the complexity of the model to prevent over-fitting [27].

3.2. Training Methodology

The network was trained with Backpropagation Through Time (BPTT) [28] and Stochastic Gradient Descent (SGD) algorithms in Tensorflow on a personal CPU system (Core i7 with 16 GB RAM). As a RNN can be seen as a normal feedforward NN with shared weights, the BPTT begins by unfolding the RNN through several steps in time. The network training then proceeds in a manner similar to training a feed-forward neural network with backpropagation, except that the training patterns were visited in a sequential order. It determines the gradients of the objective function (Equation (6)) with respect to a parameter at each time step. Finally, the BPTT calculates the gradient outputs by taking the average of the individual step-dependent gradients.

$$H_{\grave{y}}(y) = -\sum_{i} y_l \log\grave{}(y_l) \tag{6}$$

where y is the predicted probability distribution and $\grave{y}$ is the actual distribution (the one-hot vector with injury severity outcomes).

In addition, the SGD algorithm uses few examples from the input training vector and computes the outputs and the errors, and then adjusts the weights. The process is recurrent for many small sets of examples until the average of the objective function stops. The SGD method could determine a good set of weights when compared with other optimization techniques [29,30]. The training was run on a batch size 32 with 100 epochs. The best model was achieved using SGD with the decay of 0.9 and a momentum = 0.8. The best learning rate was 0.01. In addition, gradient clipping [31] with threshold 2.0 was useful to stabilize the training and to avoid gradient explosion.

3.3. Mitigating Overfitting

Networks with more complicated functions can perform better generalization performance as they learn different representations in each of their layers. However, complicated networks can easily over-fit the training data, producing poor generalization capacity and testing performance. Over-fitting occurs when a network model with high capacity fits the noise in the data instead of the underlying relationship.

Therefore, to avoid over-fitting in the proposed RNN model, three standard techniques were used. These were Gaussian noise injection into the training data [32], using a ReLU activation function in the hidden layers [20], and subsequently applying the dropout technique [29]. Dropout leads to big improvements in the prediction performance of the model. When a dropout technique is applied, the generalization capacity of the model is improved because the network is forced to learn multiple independent representations of the data. A probability of 30% was used in the dropout technique. This is because low probability has minimal effect and a high probability results in under-learning by the network.

3.4. Hyperparameter Tuning

The hyper-parameters of the RNN model were selected by performing a systematic grid search implemented in scikit-learn [33] using 100 epochs. Even though the systematic grid search requires high computational cost, better results could be obtained by systematically tuning the hyper-parameter values. Models with various combination of parameters were constructed, and a 3-fold cross-validation was used to evaluate each model. The parameters of the model with the highest validation accuracy were found to be the best parameters among the evaluated ones. Table 1 shows the optimized parameters used in the network.

Table 1. The optimized hyperparameters of the proposed RNN model.

Hyper-Parameter	Best Value	Description
Minibatch size	32	Number of training cases over which SGD update is computed.
Loss function	Categorical crossentropy	The objective function or optimization score function is also called as multiclass logloss which is appropriate for categorical targets.
Optimizer	SGD	Stochastic gradient descent optimizer.
Learning rate	0.01	The learning rate used by SGD optimizer
Gradient momentum	0.80	Gradient momentum used by SGD optimizer.
Weight decay	0.9	Learning rate decay over each update.

4. Experimental Results and Discussion

The proposed RNN model was implemented in Python using the open source TensorFlow deep learning framework developed by Google [34]. TensorFlow has automatic differentiation and parameter sharing capabilities, which allows a wide range of architectures to be easily defined and executed [34]. The proposed network was trained with 791 samples and validated with 339 samples using the Tensorflow framework. The SGD optimization algorithm was applied, with a batch size of 32 and a learning rate of 0.01.

4.1. Data

The traffic accident data for the period 2009–2015 from the North-South Expressway (NSE, Petaling Jaya, Malaysia), Malaysia were used in this study. The NSE is the longest expressway

(772 km) operated by Projek Lebuhraya Usaha Sama (PLUS) Berhad (the largest expressway operator in Malaysia, Petaling Jaya, Malaysia) and links many major cities and towns in Peninsular Malaysia. The data were obtained from the PLUS accident databases. The files used in this study were accident frequency and accident severity files in the form of an Excel spreadsheet. The accident frequency file contains the positional and descriptive accident location and the number of accidents in each road segment of 100 m. The accident records were separated according to the road bound (south, north). In contrast, the accident severity file contains the general accident characteristics such as accident time, road surface and lighting conditions, collision type, and the reported accident cause. To link the two files, the unique identity field (accident number) was used.

According to the vehicle type information in the accident data, three scenarios were found: (1) single-vehicle with object accidents, (2) two-vehicle accidents, (3) and multiple vehicle accidents (mostly three vehicles). Training a deep learning model requires many samples to capture the data structure and to avoid model overfitting. Therefore, the analysis in this study did not focus just on a single scenario, but instead, all scenarios were included in training the proposed RNN model. In total, 1130 accident records were reported during 2009–2015. Of these, 740 (approximately 65.4%) resulted in drivers damaging property only. On the other hand, 172 (15.2%) drivers were involved in possible/evident injury, and 218 (19.4%) in disabling injury.

The section of the NSE used in this study has a length of 15 km running from Ayer Keroh (210 km) to Pedas Linggi (225 km) (Figure 3). The accident severity data showed that the last section (220–225) of the NSE experienced several accidents resulting in serious injury (82) than the other sections (Table 2). Most accidents have occurred on the main route and southbound of the expressway. During the accident events, the actual accident causes were documented. The data showed that lost control, brake failure, and obstacles were the main accident causes on the NSE. With respect to lighting and surface conditions, most accidents occurred in daylight conditions and with a dry road surface. The main collision types in the accident records were out of control and rear collision. In addition, the accident time factor showed that 91.68% of the accidents occurred during the daytime. Additionally, the data also revealed that two-car accidents, single heavy car with an object and motorcycle with an object were the most recorded crashes on the NSE.

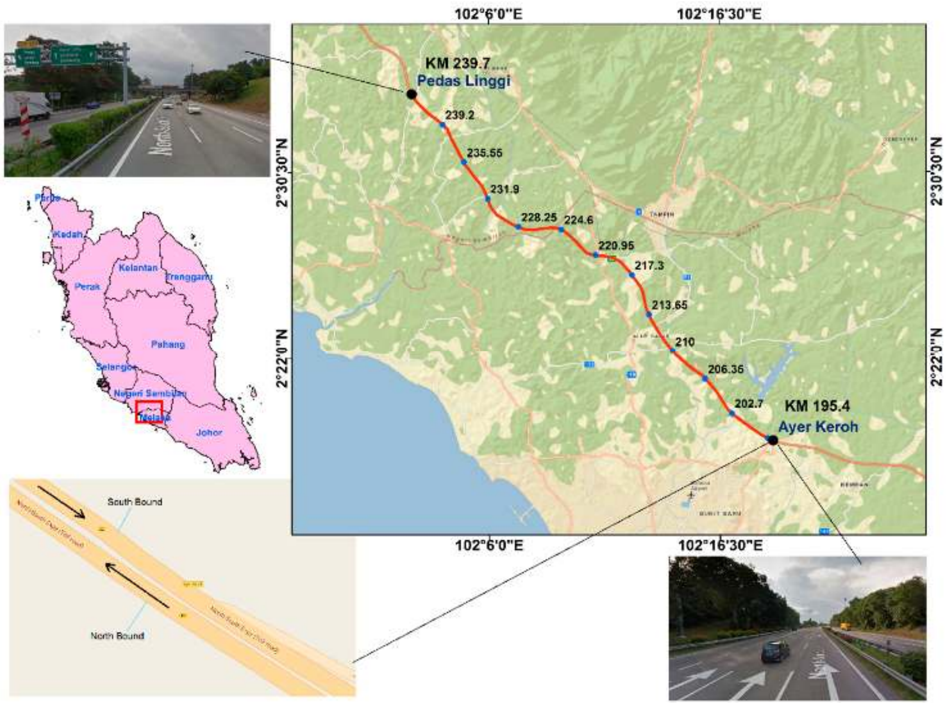

Figure 3. Location of the North-South Expressway (NSE) section analyzed in this study.

Table 2. Driver injury severity distribution according to accident related factors.

Factor	Property Damage Only	Evident Injury	Disabling Injury	Total
Location				
210–214	185	172	58	415
215–219	234	47	56	337
220–225	238	58	82	378
Road-bound				
South	453	99	139	691
North	287	73	79	439
Accident zone				
Interchange	14	3	0	17
Junction				
Lay-by	2	0	1	3
Main Route	666	155	209	1030
North Bound Entry Ramp	8	2	0	10
North Bound Exit Ramp	4	2	0	6
Rest and Service Area	21	4	2	27
South Bound Entry Ramp	2	0	1	3
South Bound Exit Ramp	7	1	3	11
Toll Plaza	16	5	2	23
Accident reported cause				
Bad Pavement Condition	0	1	0	1
Brake Failure	6	2	1	9
Bump-bump	37	12	27	76
Dangerous Pedestrian Behaviour	0	0	1	1
Drunk	0	0	1	1
Loss of Wheel	1	0	2	3
Lost control	75	18	22	115
Mechanical	5	1	0	6
Mechanical/Electrical Failure	11	0	1	12
Obstacle	43	12	6	61
Other Bad Driving	15	1	4	20
Other Human Factor/Over Load/Over Height	3	0	0	3
Over speeding	345	61	91	497
Parked Vehicle	4	4	10	18
Skidding	1	0	0	1
Sleepy Driver	134	44	42	220
Stray Animal	13	1	2	16
Tire burst	47	15	8	70
Lighting condition				
Dark with Street Light	47	6	8	61
Dark without Street Light	225	74	89	388
Dawn/Dusk	35	9	9	53
Day Light	433	83	112	628
Surface condition				
Dry	460	146	190	796
Wet	280	26	28	334
Collision type				
Angular Collision	9	2	0	11
Broken Windscreen	2	0	0	2
Cross direction	2	0	1	3
Head-on Collision	0	1	4	5
Hitting Animal	12	1	2	15
Hitting Object On Road	44	12	7	63
Others	20	0	6	26
Out of Control	457	92	107	656
Overturned	33	11	7	51
Rear Collision	137	48	81	266
Right Angle Side Collision	11	1	1	13
Side Swipe	13	4	2	19
Accident time				
Day time	677	156	203	1036
Night time	63	16	15	94

Table 2. *Cont.*

Factor	Property Damage Only	Evident Injury	Disabling Injury	Total
Vehicle type				
Car-Bus	7	3	6	16
Car-Car	499	68	60	627
Car-Heavy Car	51	11	14	76
Car-Motorcycle	4	7	22	33
Heavy Car	131	23	25	179
Heavy Car-Bus	2	3	3	8
Heavy Car-Heavy Car	24	9	15	48
Heavy Car-Motorcycle	0	1	6	7
Heavy Car-Taxi	2	0	0	2
Motorcycle	11	42	60	113
Motorcycle-Taxi	0	1	1	2
Motorcycle-Van	0	0	2	2
Taxi	1	0	1	2
Van	8	4	3	15

Before feeding the accident data into the recurrent neural network, the data must be preprocessed. The steps included were: removing missing data, data transformation, and detection of highly correlated factors. Some data were missing in the accident records; therefore, the complete raw data in which a missing data is found was removed. The data transformation included one-hot encoding for the categorical factors. In addition, correlation between predictors was assessed to detect any multicollinearity problem. First, the multiple R^2 was calculated for each factor. Second, the Variance Inflation Factor (*VIF*) was calculated from the multiple R^2 for each factor (Table 3). The highest correlation of 0.27 was found between lighting condition factor and surface condition factor. However, the highest multiple R^2 and *VIF* were found to be 0.135 and 1.156 for surface condition factor. There was no multicollinearity found among the factors if $VIF = 1.0$, however when the value exceeded 1.0, then moderate multicollinearity was found. In both cases, no high correlation was found during the model training and testing phase. Therefore, none of the factors was removed.

Table 3. Multicollinearity assessment among the accident related factors.

Factor	Multiple R^2	*VIF*
Accident location	0.062	1.066
Road bound	0.012	1.012
Accident zone	0.040	1.042
Accident reported cause	0.060	1.063
Lighting condition	0.095	1.105
Surface condition	0.135	1.156
Collision type	0.033	1.034
Accident time	0.009	1.009
Vehicle type	0.090	1.099

4.2. Results of RNN Model Performance

Figure 4 shows the accuracy performance and loss of the RNN model calculated for 100 epochs (iterations) using the training (80%) and validation (20%) datasets. In general, model accuracy on the training and validation datasets increases after each iteration with fluctuations. The fluctuations in accuracy are due to the use of the dropout technique in the model, which results in different training during every iteration. The dropout technique, which was used to prevent over-fitting, introduces some of the randomnesses to the network. In the first iteration, the accuracy was 61.28% and 66.37% for training and testing data, respectively. As the model trains during the first pass through the data, both training and validation losses decline, indicating that the model is learning the structure of the traffic accident data and possibly its temporal correlations. In the first and consecutive iterations the validation loss did not increase significantly and was always less than the training loss, indicating that the network did not overfit the training data and accurately generalized to the unseen validation data. After 100 epochs, the validation accuracy of the model was 71.77%.

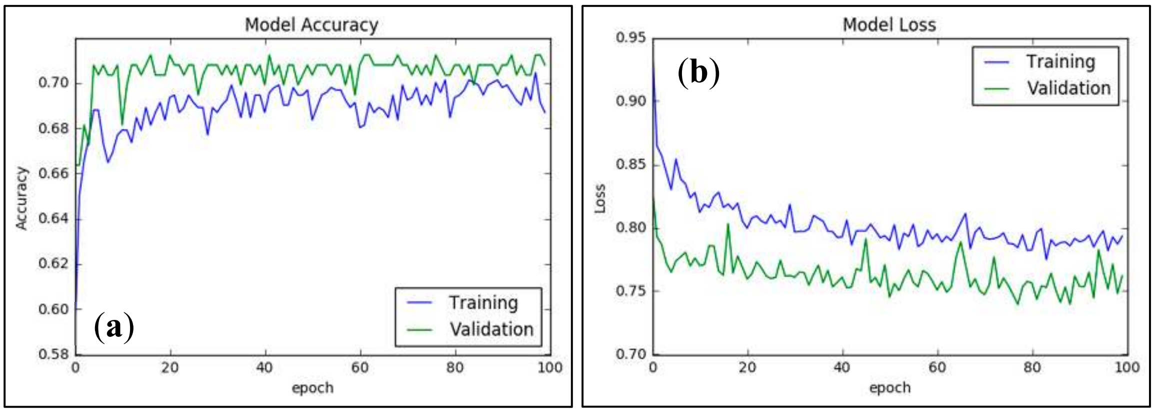

Figure 4. Accuracy performance and loss of the RNN model calculated for 100 epochs; (**a**) model accuracy; and (**b**) model loss.

4.3. Sensitivity Analysis of Optimization Algorithm

Several optimization algorithms such as SGD, Adagrad [35], RMSprop [36] and Adam [37] are available to adjust the weights and biases of a network by iterating through the training data. Each algorithm has its own advantages and disadvantages, and there are no clear guidelines for selecting an optimizer for a particular problem. Therefore, in this study, several optimization algorithms were evaluated as we searched for the best to train the proposed RNN model. The evaluated algorithms are SGD, RMSprop, Adagrad, Adadelta, Adam, Adamax, and Nadam [38]. The parameters of each algorithm were obtained from the aforementioned literature. Table 4 shows the performance of each optimization technique in predicting the severity of traffic accidents. The best validation accuracy (71.77) was achieved by the SGD method with a learning rate of 0.01. The SGD uses a small batch of the total data set to calculate the gradient of the loss function with respect to the weights and biases, using a backpropagation algorithm. The algorithm moves down the gradient by an amount proportional to its learning rate. Optimization methods such as Adam and Nadam have performed reasonably well.

Table 4. The performance of different optimization methods evaluated in this study.

Optimizer	Parameters	Training Accuracy (%)	Validation Accuracy (%)
SGD	lr = 0.01, momentum = 0.0, decay = 0.0, nesterov = False	71.79	71.77
RMSprop	lr = 0.001, rho = 0.9, epsilon = 1×10^{-8}, decay = 0.0	71.90	70.80
Adagrad	lr = 0.01, epsilon = 1×10^{-8}, decay = 0.0	71.35	71.24
Adadelta	lr = 1.0, rho = 0.95, epsilon = 1×10^{-8}, decay = 0.0	70.24	71.24
Adam	lr = 0.001, beta_1 = 0.9, beta_2 = 0.999, epsilon = 1×10^{-8}, decay = 0.0	73.12	71.68
Adamax	Lr = 0.002, beta_1 = 0.9, beta_2 = 0.999, epsilon = 1×10^{-8}, decay = 0.0	70.46	71.24
Nadam	Lr = 0.002, beta_1 = 0.9, beta_2 = 0.999, epsilon = 1×10^{-8}, schedule_decay = 0.004	74.67	71.68

4.4. Sensitivity Analysis of Learning Rate and RNN Sequence Length

Furthermore, the effect of the learning rate and RNN sequence length on the generalization ability of the proposed network in predicting the severity of traffic accidents was investigated. The learning rates varied by 0.5, 0.1, 0.05, 0.01, and 0.001. The sequence length varied by 2, 5, and 10. The choices of learning rate and sequence length were purely on an arbitrary basis. Figure 5 shows the results of these experiments. The highest validation accuracy was achieved when a learning rate of 0.01 was used. Reducing the learning rate up to 0.001 did not improve the validation accuracy, but reduced it to 70.8%. Similarly, large learning rates (e.g., 0.5) significantly reduced the performance of the network with a validation accuracy of 66.37%. On the other hand, the best validation accuracy (71.77%) was achieved with a sequence length of 2. In addition, the model performed better with a sequence length of 10 (70.87%) than 5 (70.43%). The unseen structured pattern in the model performance with different learning rates and sequence lengths indicated that the optimization of these parameters is critical to design a network that can best predict the severity of traffic accidents.

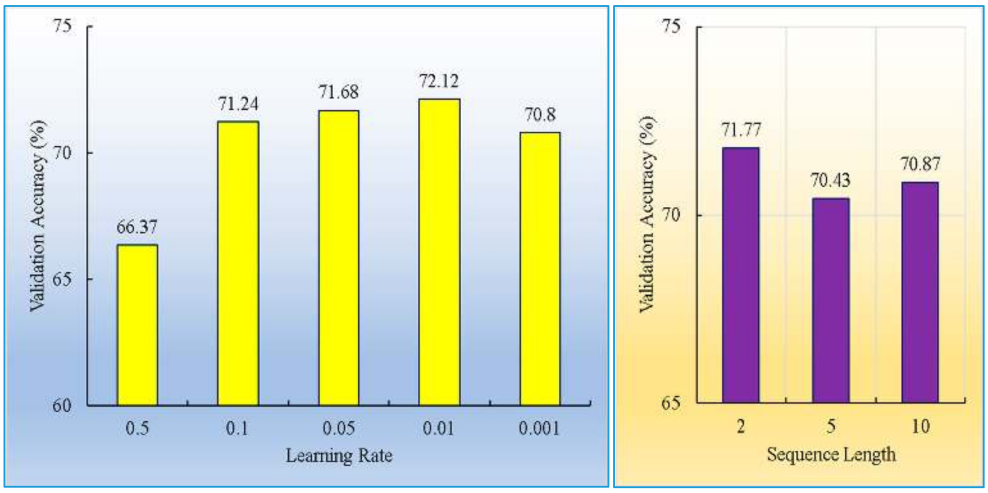

Figure 5. The sensitivity of the RNN model for different learning rate and sequence length configurations.

4.5. Network Depth Analysis

Generally, in a deep learning model, several modules and multilayers can be stacked on top of each other so that it is significant to analyze the network depth (number of hidden layers) to understand the network behavior. Table 5 shows the accuracy of the RNN model with various numbers of dense layers used on top of the LSTM layer. The best accuracy was achieved by using two dense (fully-connected) layers with 64 hidden units. This model achieved training and validation accuracies of 71.79% and 71.77% respectively. When another dense layer was added to the network, the training and validation accuracies were reduced to 70.09% and 71.24 respectively. When using five dense layers, the validation accuracy (70.35%) was found to be better than the training accuracy (67.26%); however, it could not perform better than using only two dense layers. The model started over-fitting when more than five dense layers were used. The validation accuracy of the network with eight dense layers was 56.37%, i.e., less by almost 10% than the training accuracy.

Table 5. The training and validation accuracy of the proposed RNN model with a different number of dense layers.

Number of Dense Layers	Training Accuracy (%)	Validation Accuracy (%)
2	71.79	71.77
3	70.09	71.24
5	67.26	70.35
8	65.27	56.37

In addition, the effect of the number of LSTM layers on model accuracy was assessed. Table 6 shows that the best training (71.79%) and validation (71.77%) accuracy could be achieved with a network of one LSTM layer. The accuracy was slightly reduced with another LSTM layer which was added to the network. Finally, when three LSTM layers were used, the accuracy of the model was gradually decreased and overfitted the training data.

Table 6. The training and validation accuracy of the proposed RNN model with a different number of LSTM layers.

Number of LSTM Layers	Training Accuracy (%)	Validation Accuracy (%)
1	71.79	71.77
2	71.30	71.58
3	67.11	65.34

4.6. Extraction Factor Contribution in the RNN Model

To compute the contribution of each factor in predicting the severity of traffic accidents, the profile method [39,40] was used. This technique examines the evolution of each factor with a scale of values, while the remaining factors keep their values fixed. Each factor x_i takes 11 values resulting from the division of the range, between its minimum and maximum value, into 10 equal intervals. All factors except one were initially fixed at their minimum value and then were fixed successively at their first quartile, median, third quartile and maximum values. Five values of the response variable were obtained for each 11 value and adopted by x_i, and the median of those five values was calculated. Finally, a curve with the profile of variation was obtained for every factor.

Table 7 shows the calculated weight of each factor by the profile method implemented in Python. The results indicate that the road bound and accident time have a significant effect on injury severity; the drivers had a greater risk of injuries during daytime along the southbound lanes of the NSE. Also, dry surface conditions were found to be more dangerous than a wet surface, as far as driver injury severity is concerned. During rainy times (surface is wet), drivers decrease their speed due to traffic jams, hence decreasing the severity level of driver injury. Lighting conditions, such as dark with and without street lights, increase the potential of possible injury and fatality, whereas daylight reduces the severity level of driver injury. The negative weight of the accident location factor indicates that severe accidents are most likely to happen in the section 220–225 km of the expressway. When vehicle types such as cars and motorcycles are involved in crashes, drivers are more prone to possible injury and fatality than for heavy vehicles and buses. Fatigue and speeding can give the driver a greater chance of experiencing a severe injury than other causes reported in the dataset. Accidents at the entry and exit ramps, toll plaza, and main route are more dangerous than accidents in other zones. Collision type also plays a significant role in increasing or decreasing the severity level of an accident. The analysis in this paper shows that collisions such as out of control and rear collisions increase the injury severity, whereas right angle side collision and sideswipe decrease the injury severity on the NSE.

Table 7. The calculated weights of accident related factors.

Factor	Weight
Accident location	−0.0074
Road bound	0.2899
Accident zone	0.0779
Accident reported cause	0.0469
Lighting condition	−0.0892
Surface condition	0.1785
Collision type	−0.0603
Accident time	0.3612
Vehicle type	−0.0468

4.7. Comparative Experiment

In this experiment, the proposed RNN model was compared with the traditional MLP model and the BLR approach. The advantages of MLP networks over Radial Basis Function (RBF) networks are that MLP networks are compact, less sensitive to including unnecessary inputs, and more effective for modeling data with categorical inputs.

The Grid Search algorithm was used to search the space of MLP network architectures and configurations. The number of units in the hidden layer, activation functions and the learning rate were optimized. The search space of the number of hidden units was 4 to 64 units. Five activation functions were tested such as Gaussian, Identity, Sigmoid, Softmax, and Exponential for the hidden and output layers. In addition, five values of learning rate (0.5, 0.1, 0.05, 0.01, and 0.001) were evaluated. The suboptimal network was then determined according to the best validation accuracy. The suboptimal network is the one that best represents the relationship between the input and output variables. In total, 100 network architectures with different combinations of selected parameters and network configurations were executed, and the validation accuracy on 20% of the data was computed for each created model. Then, the best network was retained. The best network had eight (8) hidden units with identity activation function, a Softmax activation in the output layer, and a learning rate of 0.01. The training and validation performance of the network was 68.90% and 65.48% respectively.

On the other hand, the BLR [41] and the computation of Markov chain Monte Carlo (MCMC) simulations were implemented in OpenBUGS software. BUGS modeling language (Bayesian Inference using Gibbs Sampling) is an effective and simplified platform to allow the computation using MCMC algorithms for all sorts of Bayesian models including BLR applied. The simulation of the posterior distribution of beta (β) allowed estimating the mean, standard deviation, and quartiles of the parameters of each explanatory variable. In the simulation stage, two MCMC chains were used to ensure convergence. The initial 100,000 iterations were discarded as burn-ins to achieve convergence and a further 20,000 iterations for each chain were performed and kept to calculate the posterior estimates of interested parameters.

In developing the Bayesian model, monitoring convergence is important because it ensures that the posterior distribution was achieved at the beginning of sampling of parameters. Convergence of multiple chains is assessed using the Brooks-Gelman-Rubin (BGR) statistic [42]. A value of less than 1.2 BGR statistic indicates convergence [42]. Convergence is also assessed by visual inspection of the MCMC trace plots for the model parameters and by monitoring the ratios of the Monte Carlo errors with respect to the corresponding standard deviation. The estimates of these ratios should be less than 0.05. In addition, MC error less than 0.05 also indicates that convergence may have been achieved [43]. The results of BLR showed training and validation accuracies of 70.30% and 58.30% respectively. In addition, the model had an average MC error of 0.047, Brooks-Gelman-Rubin (BGR) of 0.98, and model fitting performance (Deviance Information Criterion-DIC) of 322.

The comparative analysis showed that the proposed RNN model outperformed both MLP and BLR in terms of training and validation accuracies (Table 8). The BLR model performed better than MLP on the training dataset; however, its accuracy on the validation dataset was less than the MLP model. The MLP model uses only local contexts and therefore does not capture the spatial and temporal correlations in the dataset. The hidden units of the RNN model contain historical information from previous states, hence increasing the information about the data structure, which may be the main reason for its high validation accuracy over the other two methods. Building BLR models is more difficult than NN models, because they require expert domain knowledge and effective feature engineering processes. The NN model automatically captures the underlying structure of the dataset and extracts different levels of features abstractions. However, building deep learning models with various modules (e.g., fully connected networks, LSTM, convolutional layers) can be a challenging task. In addition, the BLR models are less prone to over-fitting of the training dataset than NN models, because they involve simpler relationships between the outcome and the explanatory variables. Complex networks with more hidden units and many modules often tend to over-fit more, because they detect almost any possible interaction so that the model becomes too specific to the training dataset. Therefore, optimization of network structures is very critical to avoid over-fitting and build practical prediction models. In computing the importance of an explanatory factor, the BLR models could easily calculate the factor importance and the confidence intervals of the predicted probabilities. In contrast, NN methods, which are not built primarily for statistical use, cannot easily calculate the factor importance or generate confidence intervals of the predicted probabilities unless extensive computations were done.

Table 8. Performance comparison of the proposed RNN model with Multilayer Perceptron (MLP) and Bayesian Logistic Regression (BLR) models.

Method	Training Accuracy (%)	Validation Accuracy (%)
MLP	68.90	65.48
BLR	70.30	58.30
RNN	71.79	71.77

In addition, based on the estimated weight of each accident related factor, the factors could be ranked by scaling the weights into the range of 1–100 and then giving them a rank from 1 to 9 as shown in Table 9. The three methods (MLP, BLR, and RNN) did not agree on the factors' ranking. For example, the RNN model ranked accident time as the most influential factor in injury severity, whereas accident time was ranked 2 and 7 by the BLR and MLP models, respectively. Both the RNN and BLR models agreed on several factors' ranking i.e., accident location (6), accident reported cause (5), and collision type (8). The correlation (R^2) between the RNN and BLR ranks was 0.72. In contrast, RNN and MLP did not agree on the ranking of the factors and their ranking correlation was the lowest (0.38).

Table 9. Calculated ranks of the accident related factors in the RNN, MLP, and BLR models.

Factor	MLP	BLR	RNN
Accident location	2	6	6
Road bound	8	4	2
Accident zone	5	3	4
Accident reported cause	3	5	5
Lighting condition	4	7	9
Surface condition	9	1	3
Collision type	6	8	8
Accident time	7	2	1
Vehicle type	1	9	7
BLR:MLP: R^2 = 0.51 RNN:MLP: R^2 = 0.38 MLP:BLR: R^2 = 0.51 RNN:BLR: R^2 = 0.72			

4.8. Computational Complexity of the Model

The time complexity of the RNN model was measured in terms of training and testing time per iteration. Table 10 gives information about the average training time per iteration with batch size of 32 and the average testing time per prediction. It can be seen that the model on average spends around 150 milliseconds per iteration during training and only ~13 milliseconds per prediction for new unseen examples. Although this experiment shows the computational efficiency of the model, it is also worth to note that the training time can be increased by reducing the batch size or sequence length, and also by increasing the volume of the training data.

Table 10. Average training and testing time per iteration/ prediction of the proposed model.

Time (milliseconds per iteration)	RNN Model
Training Time	150.23
Testing Time	13.17

Additionally, the major computational problem of NNs is their scalability; sometimes they become unstable when applied to larger problems. However, recent advancements in hardware and computing performance such as Graphics Processing Units (GPUs), parallel computing, and cloud computing have decreased most of the limitations of NN models including RNN-based computations.

4.9. Applicability and Limitations of the Proposed Method

The proposed RNN model is a promising traffic accident forecasting tool that has several applications in practice. First, identifying the most risky road sections (i.e., site ranking) is a daily practice of transportation agencies in many cities around the world. Accident prediction models, such as the one we proposed in this research, are often an effective solution for identifying risky road sections and helping to conduct investigations into methods for improving the safety performance of the road systems. Second, the RNN model is able to explain the underlying relationships between several accidents' related factors, such as accident time and collision type, and the injury severity outcomes. Information about the effects of accident factors on the injury severity outcomes provides huge amount of information to the transportation agencies and stakeholders. Finally, estimating the expected number of traffic accidents in a road section can help road designers to optimize the geometric alignments of the roads based on the accident scenarios.

However, the proposed RNN model has some constraints and limitations. The major limitation of the model is that the input factors are prerequisite and if any of them is missing, the output probabilities cannot be accurately estimated. Another constraint of the model is the sequence length of the RNN model, which mainly depends on the number of accident records in the training dataset. To handle this limitation, the future works should develop RNN models that operate on input sequences of variable lengths via Tensorflow dynamic calculation.

5. Conclusions

In this paper, a Recurrent Neural Network (RNN) model was developed to predict the injury severity of traffic accidents on the NSE, Malaysia. An optimized network architecture was determined through a systematic grid search for the suboptimal network hyper-parameters. Several hyper-parameters of the RNN model were critical to achieve the highest validation accuracy. The best optimization algorithm was determined to be the SGD with learning rate, momentum, and weight decay of 0.01, 0.80, and 0.9, respectively. In addition, the dropout technique helped us to reduce the complexity of the model and also the chance of over-fitting the training dataset. The RNN model achieved the best validation accuracy of 71.77% when compared to the MLP and BLR models. This indicates that additional information about temporal and contextual correlations among accident records could help the RNN model to perform better than the other models. In the sensitivity analysis of RNN sequence length, the best accuracy was achieved with a sequence length of 2, whereas the accuracy was decreased when a sequence length of 10 or 5 was used. This means that the accident data has temporal and contextual structures that could not be used by the MLP and BLR models. The impact of each factor in the RNN model was calculated using the profile method. The results showed that the most influential factors in predicting injury severity of traffic accidents are accident time and road bound. In the study area, the model predicted that the southbound section is more dangerous than the northbound, with respct to injury severity. The model also predicted drivers had a greater risk of injury on a dry surface and with lighting conditions such as dark with and without street lights.

While the proposed RNN model outperformed the MLP and BLR models and could estimate the factors' impacts, further studies should focus on developing more flexible and optimized network structures to predict accident frequency and injury severity. Future theoretical studies are encouraged to focus on improving the ability of the RNN model to represent variables and data structures, and to store data over long timescales. In addition, more applied research is recommended to develop new techniques to make use of additional information in accident datasets, such as contextual structures, spatial and temporal interactions and underlying relationships between accident factors and injury severity outcomes.

Author Contributions: Maher Ibrahim Sameen and Biswajeet Pradhan conceived and designed the experiments; Maher Ibrahim Sameen performed the experiments and analyzed the data; Biswajeet Pradhan and Maher Ibrahim Sameen contributed reagents/materials/analysis tools; Maher Ibrahim Sameen and Biswajeet Pradhan wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sameen, M.I.; Pradhan, B. Assessment of the effects of expressway geometric design features on the frequency of accident crash rates using high-resolution laser scanning data and gis. *Geomat. Nat. Hazards Risk* **2016**, 1–15. [CrossRef]
2. Pei, X.; Wong, S.; Sze, N.-N. A joint-probability approach to crash prediction models. *Accid. Anal. Prev.* **2011**, *43*, 1160–1166. [CrossRef] [PubMed]
3. Fogue, M.; Garrido, P.; Martinez, F.J.; Cano, J.-C.; Calafate, C.T.; Manzoni, P. A system for automatic notification and severity estimation of automotive accidents. *IEEE Trans. Mob. Comput.* **2014**, *13*, 948–963. [CrossRef]
4. Pawlus, W.; Reza, H.; Robbersmyr, K.G. Application of viscoelastic hybrid models to vehicle crash simulation. *Int. J. Crashworthiness* **2011**, *16*, 195–205. [CrossRef]
5. Pawlus, W.; Robbersmyr, K.G.; Karimi, H.R. Mathematical modeling and parameters estimation of a car crash using data-based regressive model approach. *Appl. Math. Model.* **2011**, *35*, 5091–5107. [CrossRef]
6. Pawlus, W.; Karimi, H.R.; Robbersmyr, K.G. Mathematical modeling of a vehicle crash test based on elasto-plastic unloading scenarios of spring-mass models. *Int. J. Adv. Manuf. Technol.* **2011**, *55*, 369–378. [CrossRef]
7. Karimi, H.R.; Pawlus, W.; Robbersmyr, K.G. Signal reconstruction, modeling and simulation of a vehicle full-scale crash test based on morlet wavelets. *Neurocomputing* **2012**, *93*, 88–99. [CrossRef]
8. Abdelwahab, H.; Abdel-Aty, M. Development of artificial neural network models to predict driver injury severity in traffic accidents at signalized intersections. *Transp. Res. Rec. J. Transp. Res. Board* **2001**, *1746*, 6–13. [CrossRef]
9. Lv, Y.; Duan, Y.; Kang, W.; Li, Z.; Wang, F.-Y. Traffic flow prediction with big data: A deep learning approach. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 865–873. [CrossRef]
10. Duan, Y.; Lv, Y.; Liu, Y.-L.; Wang, F.-Y. An efficient realization of deep learning for traffic data imputation. *Transp. Res. C* **2016**, *72*, 168–181. [CrossRef]
11. Alkheder, S.; Taamneh, M.; Taamneh, S. Severity prediction of traffic accident using an artificial neural network. *J. Forecast.* **2017**, *36*, 100–108. [CrossRef]
12. Yu, R.; Li, Y.; Shahabi, C.; Demiryurek, U.; Liu, Y. Deep learning: A generic approach for extreme condition traffic forecasting. In Proceedings of the 2017 SIAM International Conference on Data Mining (SDM), Houston, TE, USA, 27–29 April 2017.
13. Delen, D.; Sharda, R.; Bessonov, M. Identifying significant predictors of injury severity in traffic accidents using a series of artificial neural networks. *Accid. Anal. Prev.* **2006**, *38*, 434–444. [CrossRef] [PubMed]
14. Hashmienejad, S.H.-A.; Hasheminejad, S.M.H. Traffic accident severity prediction using a novel multi-objective genetic algorithm. *Int. J. Crashworthiness* **2017**, 1–16. [CrossRef]
15. Zeng, Q.; Huang, H. A stable and optimized neural network model for crash injury severity prediction. *Accid. Anal. Prev.* **2014**, *73*, 351–358. [CrossRef] [PubMed]
16. Dean, J.; Corrado, G.; Monga, R.; Chen, K.; Devin, M.; Mao, M.; Senior, A.; Tucker, P.; Yang, K.; Le, Q.V. Advances in neural information processing systems. In *Large Scale Distributed Deep Networks*; Association for Computing Machinery: New York, NY, USA, 2012; pp. 1223–1231.
17. Collobert, R.; Weston, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th International Conference on Machine learning, New York, NY, USA, 5–9 July 2008; pp. 160–167.
18. Huang, W.; Song, G.; Hong, H.; Xie, K. Deep architecture for traffic flow prediction: Deep belief networks with multitask learning. *IEEE Trans. Intell. Transp. Syst.* **2014**, *15*, 2191–2201. [CrossRef]

19. Xie, Z.; Yan, J. Detecting traffic accident clusters with network kernel density estimation and local spatial statistics: An integrated approach. *J. Transp. Geogr.* **2013**, *31*, 64–71. [CrossRef]
20. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
21. Sutskever, I.; Martens, J.; Hinton, G.E. Generating text with recurrent neural networks. In Proceedings of the 28th International Conference on Machine Learning (ICML-11), Bellevue, WA, USA, 28 June–2 July 2011; pp. 1017–1024.
22. Graves, A.; Mohamed, A.-r.; Hinton, G. Speech recognition with deep recurrent neural networks. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 6645–6649.
23. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
24. Donahue, J.; Anne Hendricks, L.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Saenko, K.; Darrell, T. Long-Term Recurrent Convolutional Networks for Visual Recognition and Description. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–15 June 2015; pp. 2625–2634.
25. Wang, X.; Abdel-Aty, M. Temporal and spatial analyses of rear-end crashes at signalized intersections. *Accid. Anal. Prev.* **2006**, *38*, 1137–1150. [CrossRef] [PubMed]
26. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NV, USA, 3–6 December 2012; pp. 1097–1105.
27. Srivastava, N.; Hinton, G.E.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
28. Hinton, G.; Rumelhart, D.; Williams, R. Learning internal representations by back-propagating errors. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*; MIT Press: London, UK, 1985; Volume 1.
29. Hinton, G.E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv* **2012**, arXiv:1207.0580.
30. Bousquet, O.; Bottou, L. The tradeoffs of large scale learning. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada; 2008; pp. 161–168.
31. Pascanu, R.; Mikolov, T.; Bengio, Y. On the difficulty of training recurrent neural networks. In Proceedings of the 30th International Conference on International Conference on Machine Learning, Atalnta, GA, USA, 16–21 June 2013; pp. 1310–1318.
32. Zur, R.M.; Jiang, Y.; Pesce, L.L.; Drukker, K. Noise injection for training artificial neural networks: A comparison with weight decay and early stopping. *Med. Phys.* **2009**, *36*, 4810–4818. [CrossRef] [PubMed]
33. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
34. Abadi, M.; Agarwal, A.; Barham, P.; Brevdo, E.; Chen, Z.; Citro, C.; Corrado, G.S.; Davis, A.; Dean, J.; Devin, M. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv* **2016**, arXiv:1603.04467.
35. Duchi, J.; Hazan, E.; Singer, Y. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* **2011**, *12*, 2121–2159.
36. Tieleman, T.; Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA Neural Netw. Mach. Learn.* **2012**, *4*, 26–31.
37. Kingma, D.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
38. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
39. Lek, S.; Belaud, A.; Dimopoulos, I.; Lauga, J.; Moreau, J. Improved estimation, using neural networks, of the food consumption of fish populations. *Oceanogr. Lit. Rev.* **1996**, *9*, 929. [CrossRef]
40. Lek, S.; Belaud, A.; Dimopoulos, I.; Lauga, J.; Moreau, J. Improved estimation, using neural networks, of the food consumption of fish populations. *Mar. Freshw. Res.* **1995**, *46*, 1229–1236. [CrossRef]
41. Xu, C.; Wang, W.; Liu, P.; Guo, R.; Li, Z. Using the bayesian updating approach to improve the spatial and temporal transferability of real-time crash risk prediction models. *Transp. Res. C* **2014**, *38*, 167–176. [CrossRef]

42. El-Basyouny, K.; Sayed, T. Application of generalized link functions in developing accident prediction models. *Saf. Sci.* **2010**, *48*, 410–416. [CrossRef]
43. Wang, C.; Quddus, M.A.; Ison, S.G. Predicting accident frequency at their severity levels and its application in site ranking using a two-stage mixed multivariate model. *Accid. Anal. Prev.* **2011**, *43*, 1979–1990. [CrossRef] [PubMed]