



International Journal of Mobile Communications

ISSN online: 1741-5217 - ISSN print: 1470-949X

<https://www.inderscience.com/ijmc>

Enhanced genetic-chaos based grasshopper optimisation algorithm for efficient crash risk prediction using novel deep learning model

D. Deva Hema, K. Ashok Kumar

DOI: [10.1504/IJMC.2025.10061199](https://doi.org/10.1504/IJMC.2025.10061199)

Article History:

Received:	20 August 2022
Last revised:	06 September 2023
Accepted:	07 October 2023
Published online:	02 December 2024

Enhanced genetic-chaos based grasshopper optimisation algorithm for efficient crash risk prediction using novel deep learning model

D. Deva Hema*

Department of Computer Science and Engineering,
SRM Institute of Science and Technology,
Ramapuram, Chennai, India
Email: devahema2010@gmail.com

*Corresponding author

K. Ashok Kumar

Department of Computer Science and Engineering,
Sathyabama Institute of Science and Technology,
Jeppiaar Nagar, Rajiv Gandhi Salai,
Chennai-600119, Tamil Nadu, India
Email: ashokkumark.csc@gmail.com

Abstract: The development of the road transport advanced collision avoidance system, which aids in accident prevention, heavily relies on accident risk prediction. The prediction of crash risk before a collision has been made possible by a number of deep learning algorithms and hybrid data driven models. In the existing convolutional-neural-network-long-short-term-memory (CNN-LSTM) model, the hyper parameters are not optimised which affects the crash risk prediction system efficiency. Therefore, to improve the collision risk prediction performance, enhanced genetic-chaos based grasshopper optimisation algorithm (EGCGOA) has been developed which optimises CNN-LSTM model's hyper parameters. CNN and LSTM are employed for data extraction and prediction respectively. The performance of EGCGOA has been evaluated with test functions, which prove the efficiency algorithm. The next generation simulation project (NGSIM) dataset was utilised to test the EGCGOA-CNN-LSTM, and it performs better than conventional techniques. The accuracy has increased which is greater than the previous model, based on the results.

Keywords: crash risk prediction; LSTM; feature extraction; reaction time; parameter optimisation.

Reference to this paper should be made as follows: Hema, D.D. and Kumar, K.A. (2025) 'Enhanced genetic-chaos based grasshopper optimisation algorithm for efficient crash risk prediction using novel deep learning model', *Int. J. Mobile Communications*, Vol. 25, No. 1, pp.84–109.

Biographical notes: D. Deva Hema received her ME in Anna University in 2007 and completed her PhD in Sathyabama Institute of Science and Technology, and working as an Assistant Professor in SRMIST in Chennai. Her research interests include the area of artificial intelligence, computer vision and deep learning.

K. Ashok Kumar received his PhD from Sathyabama University in 2016, and working as a Professor in the Department of Computer Science and Engineering in Sathyabama Institute of Science and Technology. His research interests are in grid computing, cloud computing, data analytics and web services.

1 Introduction

Traffic accidents have become a global issue due to the high number of automobiles on the road. The rear-end collision has drawn a lot of attention recently. A lot of research has been done with the advancement of self-driving vehicle technologies. The biggest cause of rear-end crashes is misjudgement of brake timing. According to a Daimler-Benz study, if the driver is informed 0.5 seconds ahead of time, 60% of collisions are avoided. Furthermore, if the motorist is notified 1.5 seconds ahead of time, 90% of collisions can be avoided (Wei et al., 2011). As a result, warning the drivers of rear-end incidents is a prominent topic among internet-of-vehicles (IoV) researchers. The National Transportation Safety Board (NTSB) reveals that collision avoidance systems can prevent 80% of deaths and injuries caused by rear end collisions (Hema and Kumar, 2021). Hence, crash risk prediction is a crucial component of collision avoidance in intelligent transportation systems (Seiler et al., 1998; Wu et al., 2014). Therefore effective crash risk prediction algorithms are required which help the drivers in making intelligent decisions to prevent from collision.

Several strategies have been developed to predict collisions and alert drivers. Time-to-collision (TTC) (Bella and Russo, 2011) is a measure that estimates time required to crash between two vehicles. The Mazda algorithm, the stop distance algorithm and Berkeley algorithms were designed to estimate the critical warning distance (Prakash et al., 2023). These algorithms estimate a warning distance depending on the gap between the front and rear vehicle. The drawback of these algorithms is that it considers only present data.

Several existing studies, applied machine learning and deep learning algorithms (Deva Hema and Ashok Kumar, 2021; Devaraj et al., 2021; Deva Hema and Kumar, 2022, 2023) in accident risk prediction which is under the domain of intelligent transportation system. Machine learning techniques handle the real-time data with historical data. Probabilistic decision-making framework (Chen et al., 2019) was developed to predict the rear end crash risk. Vehicle dynamics, Weather condition and driver response are taken as major contributory factors for collision prediction. Decisions on assisting drivers were made using the Bayesian network approach. A Kalman filter has been employed to estimate the collision probability for the stipulated interval. Pre-accident traffic dynamics were taken into account when developing random forest to forecast crash risk on motorways (Zhang et al., 2022). Support vector machine model, elastic net regularised multinomial logistic regression and conditional logistic regression (Kwak and Kho, 2016; Arbabzadeh and Jafari, 2018) models were employed for crash risk prediction. A multivariate negative binomial model (Dong et al., 2018) was proposed to predict the rear end crash risk. Xiupeng Shi proposed AutoML (Shi et al., 2020) which comprises surrogate indicators for identification of unsupervised risk, XGBoost for feature learning and Bayesian optimisation for model tuning. 91.7% of accuracy has been

achieved by AutoML model. The accuracy of safe risk identification has been achieved as 95%. Massive data handling and processing of challenging nonlinear problems are challenges for the aforementioned machine learning techniques. To solve the above mentioned problem, deep learning algorithms were developed that are highly effective in handling nonlinear complex problems with good accuracy.

Optimised back propagation neural network algorithm (Fu et al., 2020) was employed to assess the collision risk probability based on driver related factors, road condition and braking parameters. An algorithm's efficacy was verified by simulator research. The major drawback of BPNN is vanishing gradient and it does not solve temporal dependencies issues. Recurrent neural network (Tian and Pan, 2015) learns time series data in sequential manner and the same operation is performed in every sequence. Recurrent connections support preserving the information. The performance of the RNN has been affected due to vanishing gradient problems caused by long dependence. Hence, long short-term memory was designed to rectify the issue of long-term dependency. Spatial and temporal features are learnt by the LSTM model automatically so that recording of long sequences of input can be reduced. Cell state of LSTM (Morton et al., 2017) remembers the previous result and maintains input, forget and output gates which efficiently determine the useful information. LSTM-RNN (Wijnands et al., 2018) was proposed for forecasting real-time collision risk.

Convolutional neural network (CNN) model (Huang et al., 2020) was developed for collision risk forecasting. The CNN model's performance was evaluated with machine learning algorithms which resulted in high accuracy of 76%. Result depicts that deep learning algorithms have more learning capability of complex patterns than machine learning algorithms. The analysis of rear end crash risk factors has been presented using the random parameter negative binomial modelling approach (Bisht and Tiwari, 2023). But the crash risk prediction mechanism has not been implemented here. In order to effectively predict rear-end crashes, ensemble approach was developed (Taghipour et al., 2022). Before training the models, the problem of imbalanced data is solved using the synthetic minority over-sampling technique (SMOTE). Levenberg Marquardt-LSTM model (Deva Hema and Ashok Kumar, 2021) with optimised weight was proposed to forecast the collision risk. Multi-layer perceptron neural network (Ma et al., 2022) was proposed for forecasting the rear end crash risk. This model's flaw is that it requires too many parameters, which makes it ineffective. CNN-LSTM based collision risk prediction model (Li et al., 2020; Li and Abdel-Aty, 2022) was proposed where LSTM was merged with CNN to improve prediction outcome. CNN extracts the features through filters whereas LSTM provides accurate prediction results for the time series data. Result shows that the sensitivity has been increased and false alarm rate has been minimised dramatically. The drawback of CNN-LSTM model is that hyper parameters of LSTM and CNN models were not optimised which affect the efficiency of the collision risk prediction system due to high computational cost.

Several optimisation techniques (Kim and Cho, 2019; Zhang et al., 2019) have been adopted to optimise the architectural factors of the neural network. Grid search (Hinz et al., 2018) uses a predetermined range of values for hyper parameters. The implementation of grid search is simple, but computational cost is high due to exponential growth of grid. In random search, a random value has been chosen from a predefined distribution. Compared to grid search, random search is more effective and quick, but reaching optimal points requires a long time. Hence, many researchers have

combined genetic algorithms (GA) with neural network techniques to maximise the predictive model's efficacy.

Recently, several nature-inspired optimisation algorithms such as differential evolution (DE) (Zhang et al., 2019), particle swarm optimisation (PSO) (Kim and Cho, 2019), GA (Hashish et al., 2019) and bat algorithm (BA) (Yang, 2010) were developed. Unique swarm intelligence system (Saremi et al., 2017) was devised based on the grasshopper swarm's behaviour. It has been demonstrated that this method effectively resolves a variety of optimisation issues, including constrained and unconstrained issues. The frequency was optimised using a self-improved grey wolf optimisation (Rashmi and Sarvagya, 2020). Bairathi and Gopalani (2018) showed how GOA's performance has improved when compared to PSO, DE, and GA. GOA (Łukasik et al., 2017) has been used in a variety of fields due to its simplicity and effectiveness. However, GOA has certain drawbacks, including:

- 1 unbalanced exploitation and exploration processes
- 2 erratic convergence speed
- 3 a chance of slipping into the local optimum.

Chaos theory combined with GOA (CT-GOA) to increase the speed of the global convergence (Arora and Anand, 2019; Wenhan et al., 2019). To cut down on execution time, GOA is utilised in conjunction with opposition-based learning (GOA-OBL) (Bairathi and Gopalani, 2018; Ewees et al., 2018). Improved GOA was developed for the purpose of predicting short-term load (Hu et al., 2022; Wu et al., 2023). Still existing GOAs suffer with erratic convergence speed and a chance of slipping into the local optimum. Therefore EGCGOA has been developed where GOA is integrated with GA and chaotic theory. GA performs global search function from the population of solutions (Ciancio et al., 2016). GA finds set of solutions as it achieves an excellent global search. Chaotic theory increases speed of convergence. EGCGOA has a strong potential to escape from local optima while achieving faster convergence.

Summarising the above, existing research remains limited to finding the optimised hyper parameters of CNN-LSTM model for the prediction of crash risk. The performance of the LSTM network and CNN are highly based on selection of number of hidden layers, hidden units in each layer and window size. Computational cost of NN is high, when the hyper parameters have not been optimised. Therefore, there still is necessity of improvement in the development of accurate, optimised and efficient crash risk prediction systems. Therefore to enhance the crash risk prediction performance, EGCGOA has been developed. To maximise the effectiveness of the system for predicting collision risks, EGCGOA optimises the CNN-LSTM model's hyper parameters. The EGCGOA's efficiency has been evaluated with test functions which prove the efficiency of the EGCGOA algorithm. EGCGOA has a strong potential to escape from local optima while achieving faster convergence. An early collision warning system that alerts the driver can use the EGCGOA-CNN-LSTM model. The contribution of this paper is given below:

- EGCGOA is developed for the optimisation of CNN-LSTM model's hyper parameters to improve the collision risk prediction performance.

- CNN model is incorporated with LSTM model to enhance the proficiency of data extraction and prediction.
- Benchmarking models are compared to and evaluated against the proposed EGCGOA-CNN-LSTM model using the NGSIM dataset.

The organisation of the paper is given below: Section 2 provides EGCGOA-CNN-LSTM model in detail which consists of crash dataset construction, optimisation, feature extraction and prediction. Experimental design is presented in Section 3 which consists of model building and evaluation criteria are explained. Section 4 contains the results and discussions. Conclusions and future enhancements are presented in Section 5.

2 The proposed methodology

Deep learning models have been used in various areas of research for the improvement of prediction systems. CNN-LSTM based real-time collision risk prediction model produces accurate results and enhances the prediction performance. The CNN-LSTM model's disadvantage is that the hyper parameters for both the LSTM and CNN models were not optimised, which has an impact on the effectiveness of the system for predicting collision risks. Therefore to improve the collision risk prediction performance, enhanced genetic-chaos based grasshopper optimisation algorithm (EGCGOA) has been developed which optimises CNN-LSTM model's hyper parameters.

2.1 The framework of the crash risk prediction

Figure 1 shows the proposed model's framework. Initially, data is pre-processed and crash data set is constructed from the NGSIM dataset by identifying safe and crash cases based on rear end crash risk index. CNN and LSTM are employed for data extraction and prediction respectively in the proposed neural network architecture, where EGCGOA is designed for optimising the CNN-LSTM model's hyper parameters. Finally, evaluation of the proposed method has been performed with existing algorithms.

2.2 Crash data set construction

The crash dataset was constructed from the NGSIM dataset where input features are distance, velocity of subject (V_{SV}) and leading (V_{LV}) vehicle perception reaction time (PRT). The acceleration subject (α_{SV}) and leading (α_{LV}) vehicle and output feature is collision probability that indicates safe and dangerous cases. Figure 2 presents the rear end collision's car-following scenario. D_{SL} stands for the lane-ahead vehicle's distance from the subject vehicle. The possibility of a collision arises when the V_{SV} exceeds the V_{LV} or when the car in front abruptly stops. The rear end accident risk index (RCRI) is used to quantify crash risk and is a suitable surrogate measure for distinguishing between safe and dangerous scenarios (Xu et al., 2013).

$$SSD_{LV} = D + (V_{LV}^2 / (2\alpha_{LV})) \quad (1)$$

$$SSD_{SV} = (V_{SV} \times PRT) + (V_{SV}^2 / (2\alpha_{SV})) \quad (2)$$

$$RCRI = \begin{cases} 0 & \text{safe if } SSD_{LV} > SSD_{SV} \\ 1 & \text{crash case otherwise} \end{cases} \quad (3)$$

where SSD_{LV} and SSD_{SV} represent the safe stopping distance of leading vehicle and subject vehicle. D denotes clearance distance. The safe stopping distance for the leading and subject cars can be calculated using SSD_{LV} and SSD_{SV} , respectively. The safe and crash instances are divided into categories based on the RCRI.

Figure 1 Framework of the proposed model (see online version for colours)

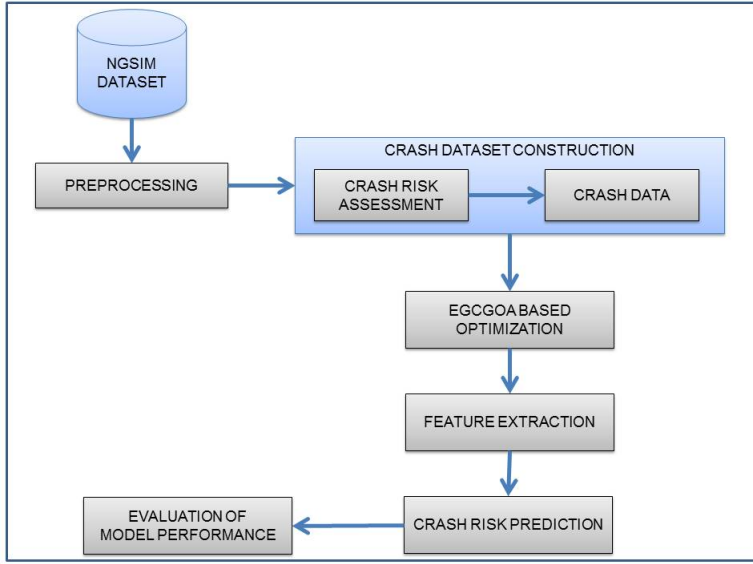
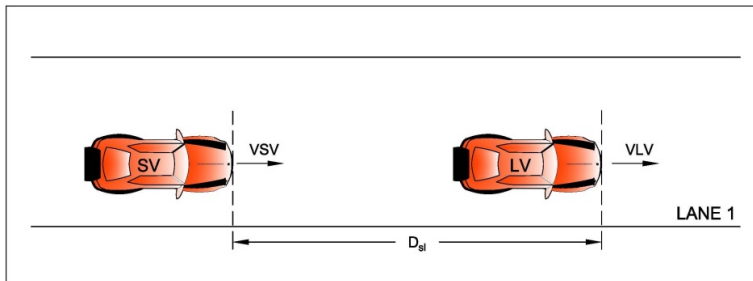


Figure 2 Car-following scenario of rear end collision (see online version for colours)



2.3 EGCGOA-based optimisation

In the proposed neural network architecture, EGCGOA has been developed and then it is combined with CNN-LSTM model for the hyper parameters optimisation of CNN-LSTM model.

2.3.1 Grasshopper optimisation algorithm

GOA is an algorithm based on characteristics of grasshopper swarms that draws inspiration from nature (Bairathi and Gopalani, 2018). The slow movement is the initial symptom of the nempth stage. The grasshopper suddenly moves when it is at the adult stage of development. Grasshopper's next trait is their constant search for food. Exploration and exploitation are the two categories of search processes that have been observed in nature-inspired algorithms. The mathematical formula for grasshopper swarming is as follows:

$$PO_i = SOI_i + GRF_i + WID_i \quad (4)$$

where PO_i denotes position i of grasshopper, SOI_i represent grasshopper's social interaction, GRF_i is of grasshopper's gravity force and WID_i wind direction respectively. SOI_i is described as follows:

$$SOI_i = \sum_{\substack{j=1 \\ j \neq i}}^N SOI(dit_{ij}) \widehat{dit_{ij}} \quad (5)$$

where dit_{ij} is the distance among the grasshoppers i and j . The definition of SOI is as follows:

$$SOI(r) = f_o e^{\frac{-r}{l}} - e^{-r} \quad (6)$$

where f_o is an intensity of attraction and l is attraction length respectively. The values of f_o and l are chosen as 1.5 and 0.5 respectively. $\widehat{dit_{ij}}$ denotes unit vector from i^{th} to j^{th} grasshopper which is stated as below:

$$\widehat{dit_{ij}} = \frac{x_j - x_i}{dit_{ij}} \quad (7)$$

GRF_i , grasshopper's gravity force is defined as follows:

$$GRF_i = -gr \widehat{e_g} \quad (8)$$

where gr and $\widehat{e_g}$ are the gravitational constant and vector of unity respectively. WID_i is computed as given below:

$$WID_i = v \widehat{e_w} \quad (9)$$

where v denotes constant drift and $\widehat{e_w}$ represents a unity vector pointing in the direction of wind. However, as mathematical approach does not provide optimised solution, the equation is modified and presented as follows:

$$X_i^d = coeff \left\{ \sum_{\substack{j=1 \\ j \neq i}}^N coeff \frac{upb_d - lrb_d}{2} s(|x_j^d - x_i^d|) \right\} + \widehat{Tg_d} \quad (10)$$

where upb_d and lrb_d denote upper bound and lower bound at dimension d respectively. $\widehat{Tg_d}$ represents target value at dimension d . $coeff$ denotes decreasing coefficient.

To improve the search process, the parameter coeff should be minimised based on the number of iterations. The coeff's formula is as follows:

$$coeff = cmai - Ite \frac{cmai - cmii}{Maxlt} \quad (11)$$

where *cmai* and *cmii* represents maximum and minimum value respectively. *Ite* and *Maxlt* denote current iteration and maximum iteration respectively.

2.3.2 Enhanced genetic-chaos based grasshopper optimisation algorithm

To enhance the convergence speed, exploration and exploitation capabilities of grasshopper optimisation algorithm (El-Shorbagy and El-Refaei, 2020), EGCGOA has been proposed. GA performs a global search function from the population of solutions (Ciancio et al., 2016). GA finds a set of solutions as it achieves an excellent global search. Chaotic theory enables to increase the speed of convergence. The EGCGOA is given in Algorithm 1.

Algorithm 1 EGCGOA algorithm

```

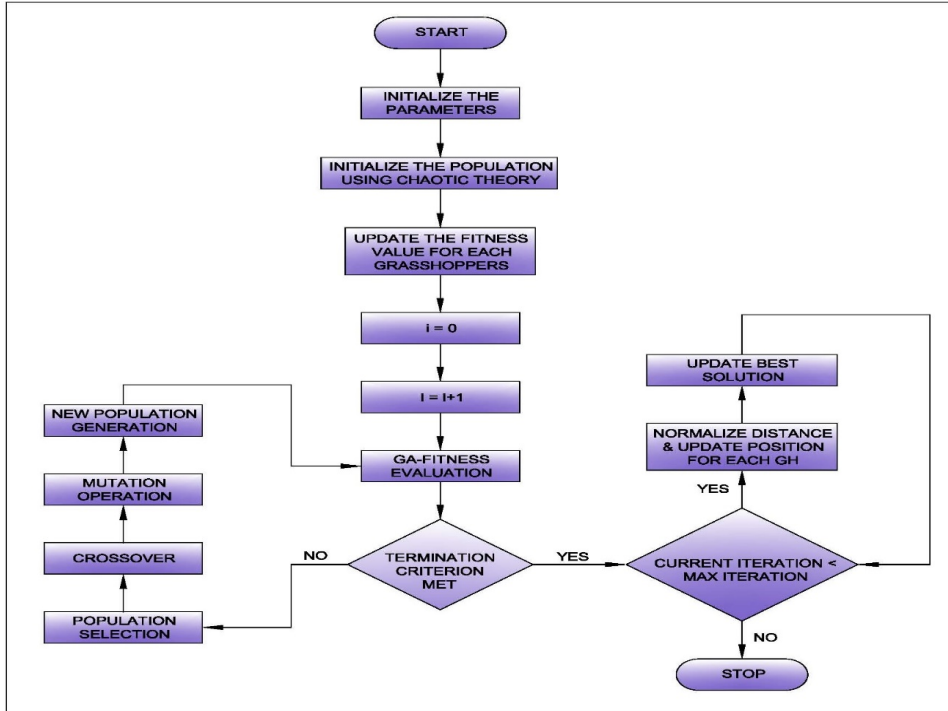
1   Initialise the parameters using chaotic theory
2   Update the fitness value for each grasshopper
3       Estimate ((TrePos + TreNeg)/(TrePos + TreNeg + FalPos + FalNeg))
4   Update the target position
5   While n = total number of generations
6       Choose the population
7       Implement a crossover procedure with a probability of 0.75.
8       Implement a crossover procedure with a probability of 0.25
9       Generate new population
10  end while
11  while(CI < MaxI)
12      Update c1 and c2
13      For each Grasshopper
14          Normalise  $dt_{ij}$ 
15          Update the position for each grasshoppers
16          Update best solution
17      end for
18  end While
19  Produce the best output.
```

The flowchart of EGCGOA is given in Figure 3. Initially, GOA generates a new population using chaotic theory. Chaos makes use of chaotic variables rather than random ones (Arora and Singh, 2017). As a result, it can execute straightforward searches at faster speeds than stochastic searches (Kohli and Arora, 2018). In the optimisation field, there are various kinds of chaotic maps (He et al., 2001; Arora and Singh, 2017). Circle GOA yields better results on a wide range of test functions than other chaotic GOAs. A

circle map is a one-dimensional map that belongs to the dynamical systems of the circles family. The following is the formula for circle map which is applied on equation (7):

$$x_{k+1} = x_k + b - \left(\frac{p}{2\pi} \right) \sin(2\pi x_k) \bmod(1) \quad (12)$$

Figure 3 Flowchart of EGCGOA model (see online version for colours)



The chaos is employed for initialisation of population, making grasshopper positions more uniform and random. Fitness value for each grasshopper has been computed to evaluate the performance of each grasshopper. GA is applied to a new population that is generated by GOA. GA's population is estimated to be 50 with a total of ten generations. Fitness of the chromosomes is evaluated based on the accuracy measure. Fitness score is defined as follows:

$$FitnessScore = ((TrePos + TreNeg) / (TrePos + TreNeg + FalPos + FalNeg)) \quad (13)$$

where *TrePos* represents true positive, *TreNeg* represents true negative, *FalPos* denotes false positive and *FalNeg* denotes false negative. Best solution is selected based on the fitness evaluation until the termination condition is met. Crossover and mutation operations are performed on the selected populations. The following definitions apply to the crossover and mutation operations:

- 1 Crossover: tournament selection is used for selecting population. Two parents $p1 = [i1, j2]$ and $p2 = [i2, j2]$ are selected. Let 'ct' be the crossover points for the parents $p1$ and $p2$. 'ct' values for the offspring are computed by:

$$ct_{new1} = (1 - \gamma)i1 + \gamma i2 \quad (14)$$

$$ct_{new2} = (1 - \gamma)i2 + \gamma i2 \quad (15)$$

where γ denotes a random number that lies between 0 and 1. Then new offsprings are generated as follows:

$$ofs1 = [ct_{new1}, j1] \quad (16)$$

$$ofs2 = [ct_{new2}, j2] \quad (17)$$

- 2 Mutation: each parent in a string $X_k \in [i_k + j_k]$ is muted with a probability 'p' by adding random values as given below:

$$X_k = \begin{cases} X_k + \Delta(t, j_k - X_k) \\ X_k + \Delta(t, X_k - i_k) \end{cases} \quad (18)$$

$$\Delta(t, y) = y \left[1 - r \left(1 - \frac{t}{T} \right)^\beta \right] \quad (19)$$

where r represents random number, β is a positive constant that is selected arbitrarily. When current iteration is less than the maximum iteration, then distance between grasshoppers is normalised and position for each grasshopper is updated. The same process is resumed until either the highest possible number of iterations has been attained or the agents have converged.

2.4 Feature extraction

CNN were developed for the purpose of learning features, which involves accepting an input. In convolutional and pooling layers (Rawat and Wang, 2017), input data is filtered and meaningful information is extracted. The convolution kernel (filter) is a small window which contains coefficient values in the form of a matrix. The convolution operation is performed to the input data to generate new features. As a result, the convolved matrix is generated which consists of convolved features. Over the retrieved features produced by the filter, the activation function rectified linear unit (ReLU) is used. A pooling layer is considered as a sub sampling technique which allows extracting significant features from convolved features and generates a lower dimension matrix. Small sliding window is utilised in a pooling layer where the sliding window takes convolved features from the convolutional layers as input and produces the output as a single value. Max pooling layer selects significant features of the convolved features. For the LSTM module to estimate crash risk, the features extracted from CNN are provided as input. Dropout process is applied for over-fitting prevention and accuracy enhancement. Dropout is a process where neuronal connections are deleted randomly.

2.5 Risk prediction

LSTM is used to predict crash events accurately. LSTM neural network learns long-term dependencies through memory cells. Memory cells in LSTM are introduced to solve the long-term dependency problem (Yuan et al., 2019). The input variables are subject

vehicle's velocity, subject's vehicle distance from leading vehicle, subject vehicle's acceleration, PRT, velocity and acceleration of the leading vehicle and output variable is collision probability. The input data matrix for crash risk prediction is represented as follows:

$$A = \begin{pmatrix} i_{11} & \cdots & i_{1n} \\ \vdots & \ddots & \vdots \\ i_{m1} & \cdots & i_{mn} \end{pmatrix} \quad (20)$$

where $i_1, i_2, i_3, \dots$ denote input variables used for crash risk prediction. 'm' and 'n' describe total samples and total independent variables.

In the LSTM unit's hidden state, the first layer input is fed as input to the second layer (Graves et al., 2013). In LSTM, the input sequence vector is mapped with output using network unit activations. The equations for input gate, output gate, memory cell and hidden state equations are given below.

$t = 1$ to 6:

$$in_t = \sigma(WT_{in}I_t + WT_{inhs}hs_{t-1} + WT_{inmc}mc_{t-1} + b_{in}) \quad (21)$$

$$fot_t = \sigma(WT_{fotin}I_t + WT_{foths}hs_{t-1} + WT_{mfot}mc_{t-1} + b_{fot}) \quad (22)$$

$$ot_t = \sigma(WT_{otin}I_t + WT_{oths}hs_{t-1} + WT_{otmc}mc_{t-1} + b_{ot}) \quad (23)$$

$$mc_t = fot_t \odot mc_{t-1} + in_t \odot \tanh(WT_{mcin}I_t + WT_{mchs}hs_{t-1} + b_{mc}) \quad (24)$$

$$hs_t = ot_t \odot \tanh(mc_t) \quad (25)$$

where in_t, fot_t, ot_t, mc_t represent value of input, forget, output and memory cell at time t respectively. WT is weight matrix, WT_{in} represents the input gate weight matrix and input 'I' at time t , it is various inputs at time t . b_{in}, b_{fo}, b_{ot} and b_{mc} represent offset of input, output and forget gates, $\tanh$ is a gating function. σ and $\odot$ represent logistic sigmoid function and element wise vector products. The values in all variables of gate vary in each time stamp. Input gate 'in' decides the amount of update needed for each unit. The forget gate 'fo' determines the information to be forgotten in each unit. Output gate 'ot' determines the state of the internal memory. The objective function is to minimise the error which is defined as follows:

$$Obj = \sum_{t=1}^m (O(t) - \widehat{PO}(t))^2 \quad (26)$$

where Obj represents objective function, $O(t)$ and $PO(t)$ are actual and predicted output at time t respectively. The algorithm for optimised crash risk prediction is given in Algorithm 1. The EGCGOA parameters are initialised and used for modifying the grasshopper's position. CNN model is trained for extracting features by performing convolution and max pooling operation. The LSTM model is trained for crash risk prediction by estimating input, forget, and output gate values, memory cell and hidden state values for each hidden layer. Finally, optimal values of number of kernels, size of kernel and pooling window size of CNN hidden layers, hidden units and window size of LSTM are derived to extract the features and to forecast the collision risk efficiently. The input from the first layer is transmitted to the second layer while the LSTM unit is in its

hidden state. The input sequence vector is mapped with output using network unit activations. Output gate ‘ot’ determines the state of the internal memory. The predicted risk is represented as follows:

$$\widehat{PO} = \widehat{PO}_1, \widehat{PO}_2, \dots, \widehat{PO}_N \quad (27)$$

PO_1, PO_2 and PO_N are the predicted risk values.

3 Experimental design

3.1 Dataset

The EGCGOA-CNN-LSTM model has been evaluated with NGSIM (Rawat and Wang, 2017) that consists of US-101 and Interstate 80 vehicle trajectory dataset from Los Angeles, California and Emeryville respectively. US-101 and I-80 vehicle trajectory dataset were collected for 45 minutes at 0.1 s interval on April 13, 2005 respectively. Six free lanes have been chosen for the study and approximately in 1,640 feet, study has happened. It is impossible to evaluate the proposed algorithm with the original dataset due to measurement errors. Therefore pre-processing the dataset is significant to extract required data from the noisy data. The data are scaled using min-max normalisation to minimise bias within a given range and undesirable columns are discarded from the dataset. Data are scaled between 0 and 1 using Min-Max normalisation which is given below.

$$Y_i = [I_i - \min(I)] / [\max(I) - \min(I)] \quad (28)$$

where I_i denotes i^{th} data point of the dataset and $\min(I)$ and $\max(I)$ represent minimum and maximum value of I . 12,053 and 12,355 samples have been taken for rear end crashes and safe cases respectively. The data is divided into two sets, a training set and a validation set, each with 80% and 20% of the total data.

3.2 Model building

The 1D kernel is used for local temporal information extraction. Two layer convolutional and pooling layers are chosen for the proposed model. ReLU is a nonlinear function which is implemented in all the convolutional layers that solves the issue of the gradient vanishing problem. The ReLU function is given below:

$$f(y) = \max(0, y) \quad (29)$$

I represent input values. Dropout rate is considered as 0.3.

Adam is used as an effective weight optimiser (Kingma and Ba, 2015). EGCGOA searches for the optimal parameters of number of convolution kernels, kernel size in each layer and pooling window size, Number of hidden layers, number of hidden units in first LSTM hidden layer, number of hidden units in second LSTM hidden layer and window size. In EGCGOA, the maximum number of iterations is considered as 10. $C_{\min} = 0.00004$, $C_{\max} = 1$, lower bound value is -1 , upper bound value is 5 , populations are initialised with the encoded bits to perform the search. The binary string structure of EGCGOA is given in Table 1. All the decimal values are converted into binary strings.

The control parameters of GA should be adjusted to estimate the fitness function. Control parameters are set from the previous studies (Kim et al., 2006) for this experiment and are detailed in Table 2.

Table 1 Binary string structure of EGCGOA

<i>Parameter</i>	<i>Range</i>
Kernels in first convolutional layer	1 to 63
Kernel size in first convolutional layer	1 to 31
Kernels in second convolutional layer	1 to 63
Kernel size in second convolutional layer	1 to 31
Pooling window size	1 to 31
LSTM hidden layers	1,2,3,4
LSTM first layer's hidden units	5 to 100
LSTM second layer's hidden units	5 to 200
Window size	1–100
Learning rate	0.00001 to 0.01

Table 2 List of control parameters

<i>Control parameters</i>	<i>Selected value</i>
Size of population	50
Total number of generations	10
Rate of crossover	0.75
Rate of mutation	0.25
Representation of chromosome	Binary string

Table 3 Test functions

<i>Name of function</i>	<i>Formula</i>	<i>Range</i>
(f1) Sphere	$f_1 = \sum_{i=1}^D x_i^2$	$[-100, 100]^D$
(f2) Rotated hyper-ellipsoid	$f_2 = \sum_{i=1}^D \sum_{j=1}^i x_j^2$	$[-65, 65]^D$
(f3) Axis parallel hyper-ellipsoid	$f_3 = \sum_{i=1}^D i x_i^2$	$[-10, 10]^D$
(f4) Schwefel	$f_4 = \sum_{i=1}^D \left(\sum_{j=1}^i x_j \right)^2$	$[-100, 100]^D$
(f5) Bent cigar	$f_5 = x_1^2 + 10^6 \sum_{i=2}^D x_i^2$	$[-10, 10]^D$

3.3 Performance evaluation index

Confusion matrix has been developed as a performance measure to assess the EGCGOA optimised CNN-LSTM model. Accuracy is a measure that describes the ratio between total number of correct predictions and the total number of records present in the dataset. The ratio of genuine positives to the total number of true positives and false positives is calculated to obtain precision. By measuring the percentage of true positives to all actual positives, recall or true positive rate is determined. The F1 score is a combination of precision and recall. Misclassification rate is calculated by dividing the total number of misclassified samples by the total number of samples. Based on the test functions described in Table 3, we assess EGCGOA's performance. Test functions are chosen to compare convergence speed of optimisation algorithms (Zhu and Wang, 2019). D represents each function's dimension. Each test function is executed ten times to determine the convergence speed.

4 Result and discussion

Tensor flow based software framework is used for the implementation of deep learning algorithms. All of the testing was performed on a laptop with the following requirements: 128 GB RAM, an Intel Core i7-8565H @ 1.8 GHz processor, and a Windows 10 operating system. Hybrid models improve the accuracy of crash risk prediction systems drastically, but neural network hyper parameters affect efficiency of the collision risk prediction system. Hence, the efficiency of the collision risk prediction system needs to be enhanced. Therefore to improve the collision risk forecasting performance, EGCGOA has been proposed to forecast the collision risk accurately with less computational cost by determining optimal topology of hybrid neural network. Then EGCGOA is integrated with CNN-LSTM model to maximise the efficiency of the collision risk prediction system. The number of kernels in convolutional layers, size of kernel and window size of pooling layer in CNN model and number of hidden layers, hidden units in each layer, window size in LSTM model are optimised using EGCGOA. 12,053 and 12,355 samples have been taken for rear end crashes and safe cases respectively.

Table 4 Optimised control parameters of CNN-LSTM model

<i>Optimised parameter</i>	<i>Value</i>
Number kernels in first convolutional layer	32
Size of kernel in first convolutional layer	23
Number kernels in second convolutional layer	64
Size of kernel in second convolutional layer	12
Pooling window size	2
Number of LSTM hidden layers	2
Hidden units in LSTM first layer	15
Hidden units in LSTM second layer	7
Window size	10
Learning rate	0.0001

Initially, GOA generates a new population using chaotic theory. Fitness value for each grasshopper has been computed to evaluate the performance of each grasshopper. GA is applied to a new population that is generated by GOA. The population size and total number of generations of GA are considered as 50 and 10 respectively. Fitness of the chromosomes is evaluated based on the accuracy measure. Best solution is selected based on the fitness evaluation until the termination condition is met. Crossover and mutation operations are performed on the selected populations. When the current iteration is less than the maximum iteration, the distance between grasshoppers is normalised and position for each grasshopper is updated. Table 4 describes optimised control parameters of CNN-LSTM hybrid model. One of the measures for evaluating the efficiency of an optimisation technique is convergence speed. Figure 4 depicts the convergence process of five variations of GOA optimisation techniques for the five different test functions from Table 3.

Table 5 Performance comparison of different GOA models with different models

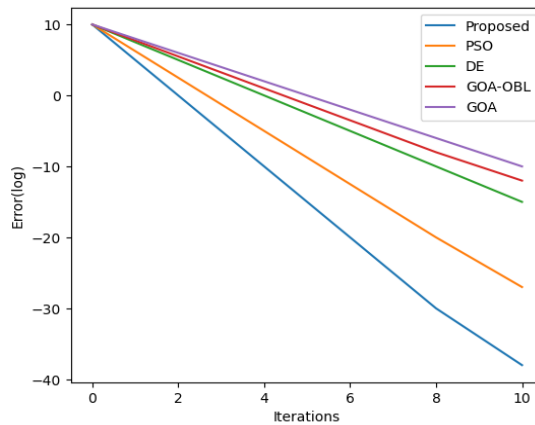
<i>Models</i>	<i>Measure</i>	<i>GOA</i>	<i>GOA-OBL</i>	<i>DE</i>	<i>PSO</i>	<i>Proposed model</i>
CNN-LSTM	Accuracy	0.952	0.96	0.966	0.972	0.985
	Precision	0.968	0.974	0.979	0.981	0.983
	Recall	0.938	0.946	0.953	0.961	0.985
	F1 Score	0.947	0.955	0.962	0.968	0.981
LM-LSTM	Accuracy	0.91	0.926	0.937	0.941	0.959
	Precision	0.92	0.919	0.919	0.92	0.925
	Recall	0.90	0.91	0.92	0.93	0.937
	F1 Score	0.902	0.907	0.912	0.917	0.905
AUTOML	Accuracy	0.898	0.902	0.907	0.908	0.917
	Precision	0.91	0.909	0.909	0.91	0.915
	Recall	0.88	0.896	0.891	0.898	0.9077
	F1 Score	0.912	0.901	0.899	0.907	0.805
CNN	Accuracy	0.888	0.892	0.897	0.898	0.9
	Precision	0.9	0.899	0.899	0.9	0.91
	Recall	0.87	0.886	0.881	0.888	0.87
	F1 Score	0.902	0.891	0.889	0.897	0.695

EGCGOA converges very faster than other algorithms over the NGSIM dataset, as shown in Figure 4. As a consequence, we can say that EGCGOA's convergence speed is the quickest for the five functions we chose. The proposed model's loss is lower in 10th iterations than in the existing models. The proposed model converges with loss reduction of 0.019. Confusion matrix has been developed as a performance measure to assess the proposed model. Table 5 discusses performance comparison of different GOA models such as GOA, GOA-OBL, DE, PSO and EGCGOA with LM-LSTM (Deva Hema and Ashok Kumar, 2021), CNN-LSTM (Li et al., 2020), AutoML (Shi et al., 2020) and CNN (Wang et al., 2020) models. Less accuracy 95.2% is achieved in the GOA-CNN-LSTM model due to unpredictable convergence speed and the potential for entering the GOA local optimum. OBL is combined with GOA to increase the speed of global convergence. Therefore accuracy has been improved to 96%. DE-CNN-LSTM has achieved 96.6% of

accuracy by expanding the local search capabilities in DE. The accuracy of PSO-CNN-LSTM has improved slightly due to its excellent features of robustness and less computational cost. EGCGOA-CNN-LSTM has achieved the highest accuracy of 98.5% due to balanced exploitation and exploration processes, stable convergence speed and very less potential for entering the local optimum. GA performs a global search function from the population of solutions and achieves an excellent global search. Chaotic theory enables us to improve the speed of convergence. Additionally, the efficacy of the proposed models is verified and evaluated against AutoML and CNN models. The training and validation accuracy of different models such as EGCGOA-CNN-LSTM, EGCGOA-LM-LSTM, EGCGOA-AutoML and EGCGOA-CNN are compared as shown in Figure 5 and it clearly demonstrates less accuracy is achieved in other state of art models. The highest accuracy of 98.5% and 97% is achieved as training and validation accuracy in the proposed model which is significantly greater accuracy than other models. The proposed model provides strong support in extracting the features, monitoring temporal properties and in predicting the crash risk with a huge volume of complicated time series crash risk data. Figure 6, Figure 7, Figure 8 and Figure 9 describe the superiority of the EGCGOA with CNN-LSTM, LMLSTM, AutoML and CNN model. Precision describes the correct classification of crash cases and recall represents the total number of correct classification of crash and safe events. More number of crash cases are identified accurately and number of correct classification of both safe and crash cases are drastically increased in the EGCGOA-CNN-LSTM model as depicted in Figure 10. The dataset is subjected to fivefold validation in order to assess the collision risk prediction system's efficacy.

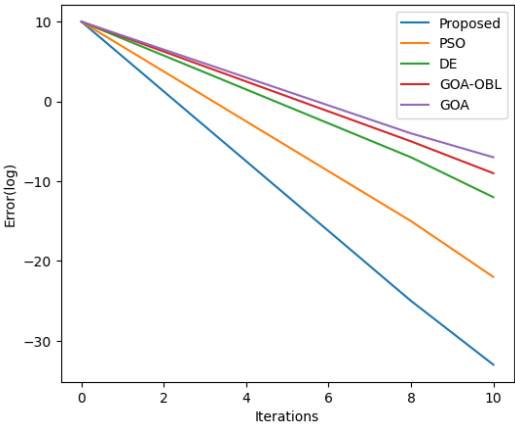
Figure 11 and Figure 12 present execution time required for training and testing dataset using different optimisation algorithms where training and testing data are divided into five folds. The proposed model takes less execution time compared to other optimisation algorithms. The receiver operating characteristic curve's two-dimensional areas under the curve are represented by area under the curve, or AUC. Higher AUC improves the collision risk prediction system's efficiency and makes it possible to accurately predict crash cases. Figure 13 show that the proposed model achieves the greatest AUC value of 0.987 when compared to other conventional algorithms.

Figure 4 Convergence curves, (a) f_1 , (b) f_2 , (c) f_3 , (d) f_4 , (e) f_5 (see online version for colours)

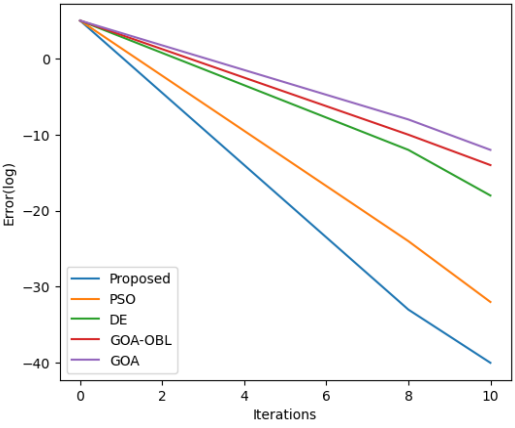


(a)

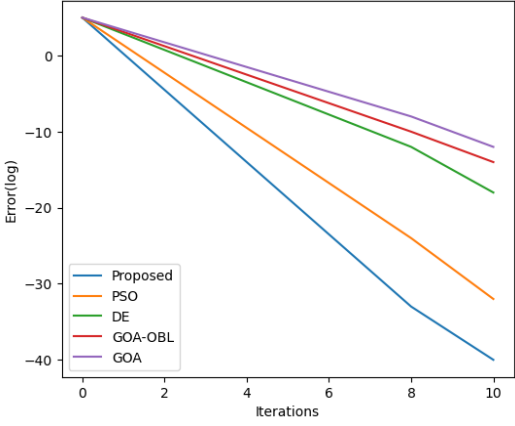
Figure 4 Convergence curves, (a) $f1$, (b) $f2$, (c) $f3$, (d) $f4$, (e) $f5$ (continued) (see online version for colours)



(b)



(c)



(d)

Figure 4 Convergence curves, (a) f_1 , (b) f_2 , (c) f_3 , (d) f_4 , (e) f_5 (continued) (see online version for colours)

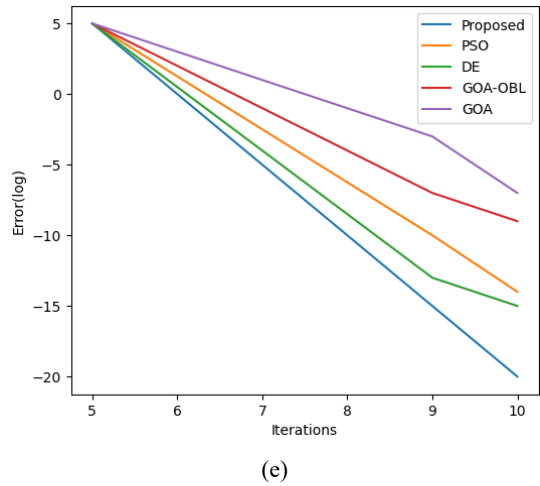


Figure 5 Training and validation accuracy of different models (see online version for colours)

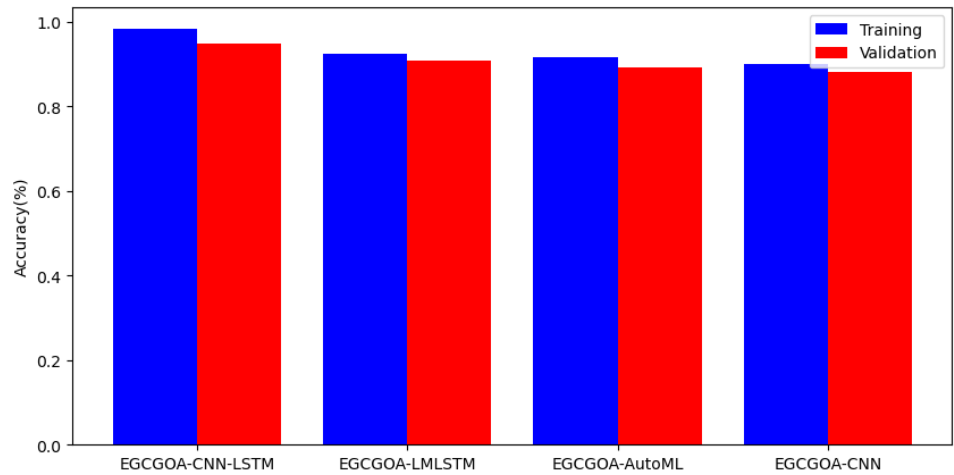
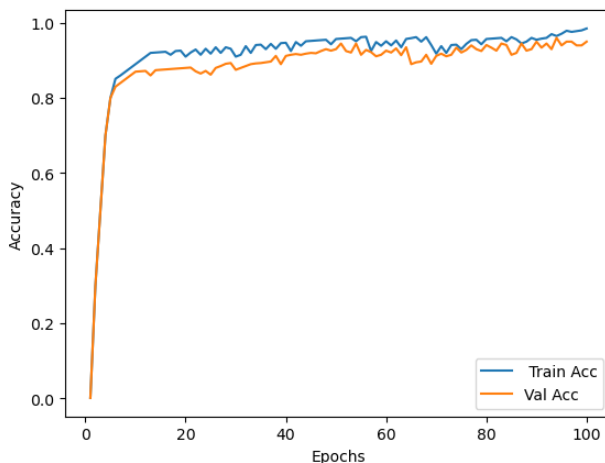
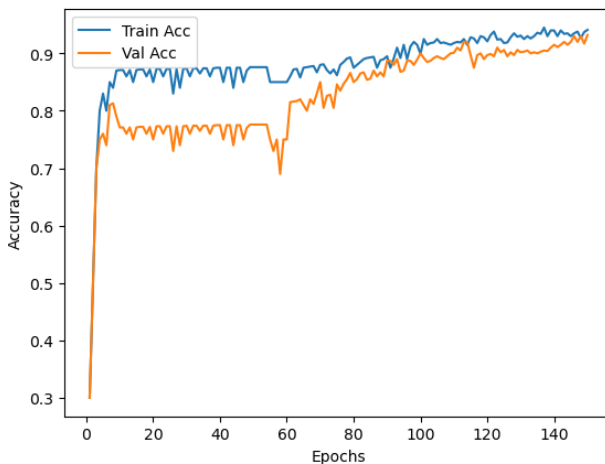


Table 6 The efficiency comparison

Model	Learning rate	False alarm rate	Missed alarm rate
EGCGOA-CNN-LSTM	0.0001	0.006	0.008
EGCGOA-CNN-LSTM	0.001	0.0211	0.0101
EGCGOA-LMLSTM	0.001	0.0326	0.0111
EGCGOA-AUTOML	0.001	0.092	0.028
EGCGOA-CNN	0.001	0.097	0.029

Figure 6 Training and validation accuracy of the EGCGOA-CNN-LSTM (see online version for colours)**Figure 7** Training and validation accuracy of the EGCGOA-LM-LSTM (see online version for colours)

The proposed model identifies impending crashes accurately and presents warnings to the drivers. Table 6 describes efficiency comparison of various models where the EGCGOA-CNN-LSTM enables false and missed alarm rate to be minimised drastically. The number of false negatives and false positives cases are significantly reduced in the EGCGOA-CNN-LSTM model.

Figure 8 Training and validation accuracy of the EGCGOA-AUTOML (see online version for colours)

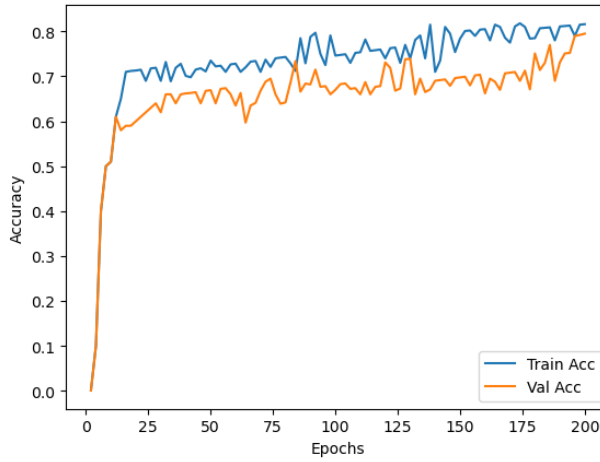
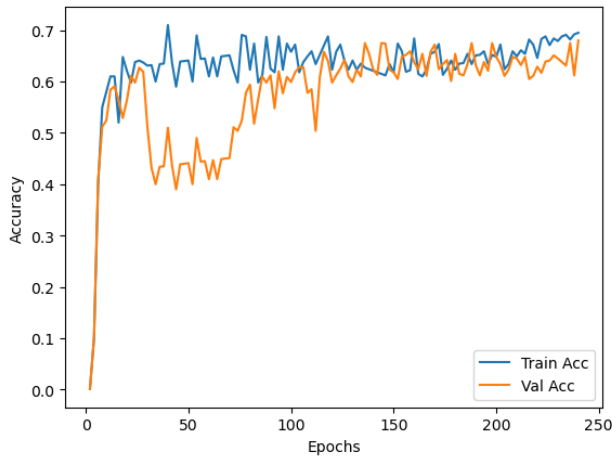


Figure 9 Training and validation accuracy of the EGCGOA-CNN (see online version for colours)



McNemar's test is a non-parametric test which is used for performance comparison of the CNN-LSTM and EGCGOA-CNN-LSTM model. McNemar statistical test finds significant changes between two models. The predictive performance between models is compared at 1% significance level. The p-value is 0.00329 which is less than α value 0.05. The test result reveals that EGCGOA-CNN-LSTM performs better than CNN-LSTM model significantly.

Figure 10 Comparison of precision and recall (see online version for colours)

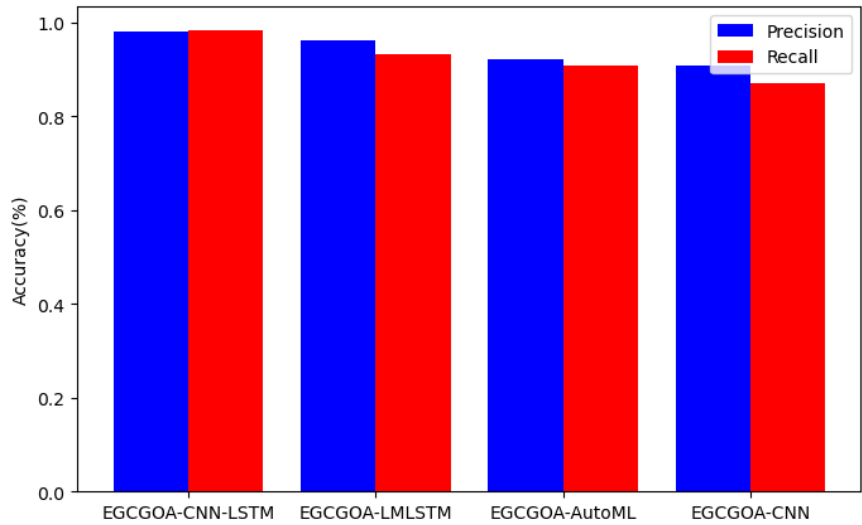


Figure 11 Execution Training time of different models (see online version for colours)

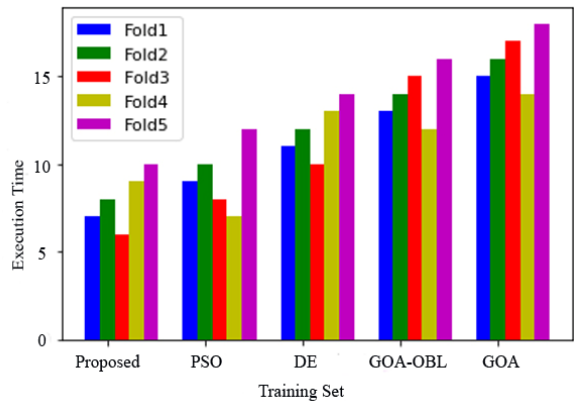


Figure 12 Execution testing time of different models (see online version for colours)

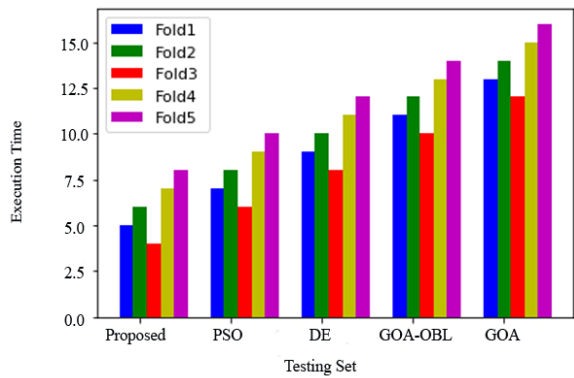
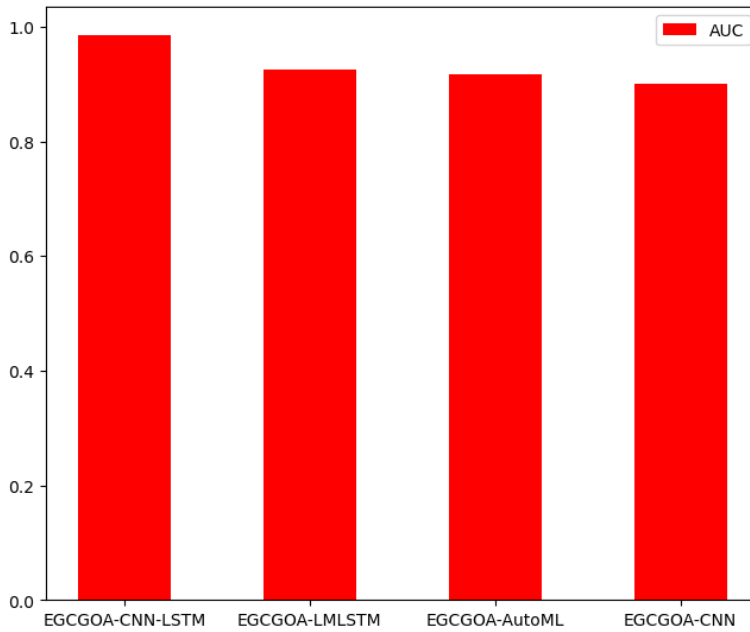


Figure 13 The comparison of AUC for different algorithms (see online version for colours)

5 Conclusions

This paper introduced EGCGOA to enhance the crash risk prediction performance. To increase the effectiveness of the collision risk prediction system, EGCGOA adjusts the CNN-LSTM model's hyper parameters. EGCGOA has a strong potential to escape from local optima while achieving faster convergence. LSTM and CNN parameters should be adjusted to boost the prediction performance of the crash risk. CNN-LSTM model's hyper parameters are optimised by EGCGOA. Initially, Performance Comparison of different models such as GOA, GOA-OBL, DE, PSO and EGCGOA with CNN-LSTM, AutoML and CNN classifier has been performed. EGCGOA-CNN-LSTM has achieved the highest accuracy of 98.5% to forecast the crash risk. The result describes that EGCGOA optimised CNN-LSTM model enhances the prediction performance of the crash risk. In addition, EGCGOA converges very faster than other optimisation algorithms. In contrast to the current CNN-LSTM, where the false and missed alarm rates are as high as 21% and 10% respectively, the proposed model significantly reduces these rates to 0.6% and 0.8%. Therefore the proposed model predicts the impending crashes accurately and warns the drivers. The NGSIM dataset lacks environmental and vehicle conditions. Therefore in future, augmentation of environmental and vehicle conditions with NGSIM data will enhance the prediction performance. In addition, weather conditions can be considered for the crash risk prediction in the future.

References

- Arbabzadeh, N. and Jafari, M. (2018) 'A data-driven approach for driving safety risk prediction using driver behavior and roadway information data', *IEEE Transactions on Intelligent Transportation Systems*, Vol. 19, No. 2, pp.446–460, <https://doi.org/10.1109/TITS.2017.2700869>.
- Arora, S. and Anand, P. (2019) 'Chaotic grasshopper optimization algorithm for global optimization', *Neural Computing and Applications*, Vol. 31, No. 8, <https://doi.org/10.1007/s00521-018-3343-2>.
- Arora, S. and Singh, S. (2017) 'An improved butterfly optimization algorithm with chaos', *Journal of Intelligent & Fuzzy Systems*, Vol. 32, No. 1, pp.1079–1088, <https://doi.org/10.3233/JIFS-16798>.
- Bairathi, D. and Gopalani, D. (2018) 'An improved opposition based grasshopper optimisation algorithm for numerical optimization', in *International Conference on Intelligent Systems Design and Applications*, pp.843–851, Springer, https://doi.org/10.1007/978-3-030-16660-1_82.
- Bella, F. and Russo, R. (2011) 'A collision warning system for rear-end collision: a driving simulator study', *Procedia – Social and Behavioral Sciences*, Vol. 20, pp.676–686, <https://doi.org/10.1016/j.sbspro.2011.08.075>.
- Bisht, L.S. and Tiwari, G. (2023) 'Assessment of fatal rear-end crash risk factors of an expressway in india: a random parameter NB modeling approach', *Journal of Transportation Engineering, Part A: Systems*, Vol. 149, No. 1, p.4022111, <https://doi.org/10.1061/JTEPBS.000076>.
- Chen, C., Liu, X., Chen, H.H., Li, M. and Zhao, L. (2019) 'A rear-end collision risk evaluation and control scheme using a bayesian network model', *IEEE Transactions on Intelligent Transportation Systems*, Vol. 20, No. 1, pp.264–284, <https://doi.org/10.1109/TITS.2018.2813364>.
- Ciancio, C., Ambrogio, G., Gagliardi, F. and Musmanno, R., (2016) 'Heuristic techniques to optimize neural network architecture in manufacturing applications', *Neural Computing and Applications*, Vol. 27, No. 7, pp.2001–2015, <https://doi.org/10.1007/s00521-015-1994-9>.
- Deva Hema, D. and Ashok Kumar, K. (2021) 'Levenberg-Marquardt-LSTM based Efficient rear-end crash risk prediction system optimization', *International Journal of Intelligent Transportation Systems Research*, Vol. 20, pp.132–141, <https://doi.org/10.1007/s13177-021-00273-2>.
- Deva Hema, D. and Kumar, A. (2022) 'Novel algorithm for multivariate time series crash risk prediction using CNN-ATT-LSTM model', *Journal of Intelligent & Fuzzy Systems*, Vol. 43, No. 4, pp.4201–4213, <https://doi.org/10.3233/JIFS-211775>.
- Deva Hema, D. and Kumar, K.A. (2023) 'Optimized deep neural network based intelligent decision support system for traffic state prediction', *International Journal of Intelligent Transportation Systems Research*, Vol. 21, No. 1, pp.26–35.
- Devaraj, D., Chandrasekaran, D., Pandian, B., Binitha, B., Dipikhasre, M. and Anil, R. (2021) 'Road accident analysis in Kerala and location based severity level classification using decision tree algorithm', *Paideuma Journal*, Vol. 14, No. 14, pp.41–50.
- Dong, C., Shao, C., Li, J. and Xiong, Z. (2018) 'An improved deep learning model for traffic crash prediction', *Journal of Advanced Transportation*, Vol. 2018, pp.1–13, <https://doi.org/10.1155/2018/3869106>.
- El-Shorbagy, M.A. and El-Refaey, A.M. (2020) 'Hybridization of grasshopper optimization algorithm with genetic algorithm for solving system of non-linear equations', *IEEE Access*, Vol. 8, pp.220944–220961, <https://doi.org/10.1109/ACCESS.2020.3043029>.
- Ewees, A.A., Abd Elaziz, M. and Houssein, E.H. (2018) 'Improved grasshopper optimization algorithm using opposition-based learning', *Expert Systems with Applications*, Vol. 112, <https://doi.org/10.1016/j.eswa.2018.06.023>.

- Fu, Y., Li, C., Luan, T.H., Zhang, Y. and Yu, F.R. (2020) 'Graded warning for rear-end collision: an artificial intelligence-aided algorithm', *IEEE Transactions on Intelligent Transportation Systems*, Vol. 21, No. 2, pp.565–579, <https://doi.org/10.1109/TITS.2019.2897687>.
- Graves, A., Mohamed, A. and Hinton, G. (2013) 'Speech recognition with deep recurrent neural networks', in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, pp.6645–6649, <https://doi.org/10.1109/ICASSP.2013.6638947>.
- Hashish, I.A., Forni, F., Andreotti, G., Facchinetti, T. and Darjani, S. (2019) 'A hybrid model for bitcoin prices prediction using hidden Markov models and optimized LSTM networks', in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, IEEE, pp.721–728, <https://doi.org/10.1109/ETFA.2019.8869094>.
- He, D., He, C., Jiang, L.G., Zhu, H.W. and Hu, G.R. (2001) 'Chaotic characteristics of a one-dimensional iterative map with infinite collapses', *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, Vol. 48, No. 7, pp.900–906, <https://doi.org/10.1109/81.933333>.
- Hema, D.D. and Kumar, K.A. (2021) 'Hyperparameter optimization of LSTM based driver's aggressive behavior prediction model', in *International Conference on Artificial Intelligence and Smart Systems (ICAIS 2021)*, IEEE, Coimbatore, pp.751–756, <https://doi.org/10.1109/ICAIS50930.2021.9396047>.
- Hinz, T., Navarro-Guerrero, N., Magg, S. and Wermter, S. (2018) 'Speeding up the hyperparameter optimization of deep convolutional neural networks', *International Journal of Computational Intelligence and Applications*, Vol. 17, No. 2, p.1850008, <https://doi.org/10.1142/S1469026818500086>.
- Hu, H., Xia, X., Luo, Y., Zhang, C., Nazir, M.S. and Peng, T. (2022) 'Development and application of an evolutionary deep learning framework of LSTM based on improved grasshopper optimization algorithm for short-term load forecasting', *Journal of Building Engineering*, Vol. 57, p.104975, <https://doi.org/10.1016/j.job.2022.104975>.
- Huang, T., Wang, S. and Sharma, A. (2020) 'Highway crash detection and risk estimation using deep learning', *Accident Analysis & Prevention*, Vol. 135, p.105392, <https://doi.org/10.1016/j.aap.2019.105392>.
- Kim, M-J., Min, S-H. and Han, I. (2006) 'An evolutionary approach to the combination of multiple classifiers to predict a stock price index', *Expert Systems with Applications*, Vol. 31, No. 2, pp.241–247, <https://doi.org/10.1016/j.eswa.2005.09.020>.
- Kim, T-Y. and Cho, S-B. (2019) 'Particle swarm optimization-based CNN-LSTM networks for forecasting energy consumption', in *2019 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, pp.1510–1516, <https://doi.org/10.1109/CEC.2019.8789968>.
- Kingma, D.P. and Ba, J. (2015) 'Adam: a method for stochastic optimization', in *3rd International Conference for Learning Representations*, San Diego, pp.1–15, <http://arxiv.org/abs/1412.6980>.
- Kohli, M. and Arora, S. (2018) 'Chaotic grey wolf optimization algorithm for constrained optimization problems', *Journal of Computational Design and Engineering*, Vol. 5, No. 4, <https://doi.org/10.1016/j.jcde.2017.02.005>.
- Kwak, H-C. and Kho, S. (2016) 'Predicting crash risk and identifying crash precursors on Korean expressways using loop detector data', *Accident Analysis & Prevention*, Vol. 88, pp.9–19, <https://doi.org/10.1016/j.aap.2015.12.004>.
- Li, P. and Abdel-Aty, M. (2022) 'Real-time crash likelihood prediction using temporal attention-based deep learning and trajectory fusion', *Journal of Transportation Engineering, Part A: Systems*, Vol. 148, No. 7, <https://doi.org/10.1061/JTEPBS.0000697>.
- Li, P., Abdel-Aty, M. and Yuan, J. (2020) 'Real-time crash risk prediction on arterials based on LSTM-CNN', *Accident Analysis & Prevention*, Vol. 135, p.105371, <https://doi.org/10.1016/j.aap.2019.105371>.
- Łukasik, S., Kowalski, P.A., Charytanowicz, M. and Kulczycki, P. (2017) 'Data clustering with grasshopper optimization algorithm', in *2017 Federated Conference on Computer Science and Information Systems (FedCSIS)*, IEEE, pp.71–74, <https://doi.org/10.15439/2017F340>.

- Ma, X., Yu, Q. and Liu, J. (2022) 'Modeling urban freeway rear-end collision risk using machine learning algorithms', *Sustainability*, Vol. 14, No. 19, p.12047, <https://doi.org/10.3390/su141912047>.
- Morton, J., Wheeler, T.A. and Kochenderfer, M.J. (2017) 'Analysis of recurrent neural networks for probabilistic modeling of driver behavior', *IEEE Transactions on Intelligent Transportation Systems*, Vol. 18, No. 5, pp.1289–1298, <https://doi.org/10.1109/TITS.2016.2603007>.
- Prakash, T.S., Kumar, A.S., Durai, C.R.B. and Ashok, S. (2023) 'Enhanced Elman spike neural network optimized with flamingo search optimization algorithm espoused lung cancer classification from CT images', *Biomedical Signal Processing and Control*, Vol. 84, p.104948.
- Rashmi, N. and Sarvagya, M. (2020) 'Self-improved grey wolf optimization for estimating carrier frequency offset in SCM-OFDM systems', *International Journal of Pervasive Computing and Communications*, Vol. 16, No. 1, pp.53–73, <https://doi.org/10.1108/IJPC-03-2019-0020>.
- Rawat, W. and Wang, Z. (2017) 'Deep convolutional neural networks for image classification: a comprehensive review', *Neural Computation*, Vol. 29, No. 9, pp.2352–2449, https://doi.org/10.1162/neco_a_00990.
- Saremi, S., Mirjalili, S. and Lewis, A. (2017) 'Grasshopper optimisation algorithm: theory and application', *Advances in Engineering Software*, Vol. 105, pp.30–47, <https://doi.org/10.1016/j.advengsoft.2017.01.004>.
- Seiler, P., Song, B. and Hedrick, J.K. (1998) 'Development of a collision avoidance system', *Journal of Passenger Cars*, Vol. 107, pp.1334–1340.
- Shi, X., Wong, Y.D., Chai, C. and Li, M.Z.F. (2020) 'An automated machine learning (AutoML) method of risk prediction for decision-making of autonomous vehicles', *IEEE Transactions on Intelligent Transportation Systems*, pp.1–10, <https://doi.org/10.1109/TITS.2020.3002419>.
- Taghipour, H., Parsa, A.B., Chauhan, R.S., Derrible, S. and Mohammadian, A.K. (2022) 'A novel deep ensemble based approach to detect crashes using sequential traffic data', *IATSS Research*, Vol. 46, No. 1, pp.122–129, <https://doi.org/10.1016/j.iatssr.2021.10.004>.
- Tian, Y. and Pan, L. (2015) 'Predicting short-term traffic flow by long short-term memory recurrent neural network', in *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, IEEE, pp.153–158, <https://doi.org/10.1109/SmartCity.2015.63>.
- Wang, X., Liu, J., Qiu, T., Mu, C., Chen, C. and Zhou, P. (2020) 'A real-time collision prediction mechanism with deep learning for intelligent transportation system', *IEEE Transactions on Vehicular Technology*, Vol. 69, No. 9, pp.9497–9508, <https://doi.org/10.1109/TVT.2020.3003933>.
- Wei, Z., Xiang, S., Xuan, D. and Xu, L. (2011) 'An adaptive vehicle rear-end collision warning algorithm based on neural network', in *International Conference on Information and Management Engineering ICCIC 2011*, Springer, Berlin, Heidelberg, pp.305–314, https://doi.org/10.1007/978-3-642-24097-3_46.
- Wenhan, X., Yuanxing, W., Di, Q. and Daneshvar Rouyendegh, B. (2019) 'Improved grasshopper optimization algorithm to solve energy consuming reduction of chiller loading', *Energy Sources, Part A: Recovery, Utilization, and Environmental Effects* [Preprint], <https://doi.org/10.1080/15567036.2019.1687622>.
- Wijnands, J.S., Thompson, J., Aschwanden, G.D. and Stevenson, M. (2018) 'Identifying behavioural change among drivers using long short-term memory recurrent neural networks', *Transportation Research Part F: Traffic Psychology and Behaviour*, Vol. 53, pp.34–49, <https://doi.org/10.1016/j.trf.2017.12.006>.
- Wu, C., Peng, L., Huang, Z., Zhong, M. and Chu, D. (2014) 'A method of vehicle motion prediction and collision risk assessment with a simulated vehicular cyber physical system', *Transportation Research Part C: Emerging Technologies*, Vol. 47, pp.179–191, <https://doi.org/10.1016/j.trc.2014.07.002>.

- Wu, L., Wu, J. and Wang, T. (2023) 'Enhancing grasshopper optimization algorithm (GOA) with levy flight for engineering applications', *Scientific Reports*, Vol. 13, No. 1, p.124, <https://doi.org/10.1038/s41598-022-27144-4>.
- Xu, C., Liu, P., Wang, W. and Jiang, X. (2013) 'Development of a crash risk index to identify real time crash risks on freeways', *KSCE Journal of Civil Engineering*, Vol. 17, No. 7, pp.1788–1797, <https://doi.org/10.1007/s12205-013-0353-6>.
- Yang, X-S. (2010) 'A new metaheuristic bat-inspired algorithm', in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, Springer, pp.65–74, https://doi.org/10.1007/978-3-642-12538-6_6.
- Yuan, J., Abdel-Aty, M., Gong, Y. and Cai, Q. (2019) 'Real-time crash risk prediction using long short-term memory recurrent neural network', *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2673, No. 4, pp.314–326, <https://doi.org/10.1177/0361198119840611>.
- Zhang, S., Chen, Y., Huang, X. and Cai, Y. (2019) 'Text classification of public feedbacks using convolutional neural network based on differential evolution algorithm', *International Journal of Computers Communications & Control*, Vol. 14, No. 1, pp.124–134, <https://doi.org/10.15837/ijccc.2019.1.3420>.
- Zhang, Z., Nie, Q., Liu, J., Hainen, A., Islam, N. and Yang, C. (2022) 'Machine learning based real-time prediction of freeway crash risk using crowdsourced probe vehicle data', *Journal of Intelligent Transportation Systems*, pp.1–19, <https://doi.org/10.1080/15472450.2022.2106564>.
- Zhu, X. and Wang, N. (2019) 'Cuckoo search algorithm with onlooker bee search for modeling PEMFCs using T2FNN', *Engineering Applications of Artificial Intelligence*, Vol. 85, pp.740–753, <https://doi.org/10.1016/j.engappai.2019.07.019>.