# Analysis of Vehicle Collision Prediction Algorithms Using CNN

Tanya Jain*, Garima Aggarwal and Sumita Gupta

**Abstract** In today's driven world, Vehicle Collision(VC) is one of the primary causes of injuries and fatalities on the road. The recent advances in technology help us to predict and potentially avoid such incidents for a safer and smarter traveling experience. Thus, there is also a need to evaluate, compare and improve on these technologies. This paper includes analysis of 108 convolution neural networks(CNN) created with different permutations of configurations (config.): Gaussian Mixture Model, kaiming weights and biases, average or max pooling, dropout and additional fully connected layer, negative log likelihood loss, cross entropy loss or multi-class hinge loss, stochastic gradient descent or Adam's optimizer, and padding in convolution layers. The detection of VCs is performed upon 8,284 data points using CNN. The analysis of best and worst performing CNNs has also been presented to understand the nature of the prediction resulting due to certain pairings. The major contribution of this paper involves the proposal of a collision detection system which is highly efficient, accurate and loss-less with low computation cost in memory and time, making it implementable in applications requiring less infrastructure. It also analyses the different config. that work for this task of detecting collisions.

## 1 Introduction

Introduction of autonomous vehicles (AV) can reduce maneuvers of conventional vehicle (CV) drivers like 'right of way violation' [1] and accidents with pedestrian by 90% [2]. Yet, AVs increase CV drivers' maneuvers of 'following too closely' and 'unsafe speed' [1]. Though AVs perform better in structured environments over complex ones[9], AVs can fail in right decision making despite testing in highly controlled settings. To prevent unforeseen accidents and improve the AVs' efficiency and safety, research and development is required on VC and its impact.

Traditional machine learning algorithms [13] have been used to detect the accident via computer vision techniques such as random forest classifiers (RFC) [8],

Tanya Jain*
Department of CSE, Amity University, Sector-125 Noida, Uttar Pradesh, India, +918800967475, e-mail: tanyaacjain@tanya-jain.xyz

Garima Aggarwal
Department of CSE, Amity University, Sector-125 Noida, Uttar Pradesh, India, +918750000840, e-mail: gmehta@amity.edu

Sumita Gupta
Department of CSE, Amity University, Sector-125 Noida, Uttar Pradesh, India, +919971503555, e-mail: sgupta4@amity.edu

support vector machines (SVM) and artificial neural networks (ANNs). Traditional feature extraction methods like local binary pattern[3], histogram of gradients[18], maximally stable extremal regions[6] and speeded-up robust features [7] have been used on the RFC to extract image features in a single matrix. The proposed model (PM) is inspired from using MaxPooling (MaxPool) to select the local maxima from multiple input layers and generate the desired features in the output layer [14]. The ResNet50 model was used in [10] for collision detection. Other much extensive work include using the LSTM architecture integrated along with Augmented Context Mining (ACM) into the Faster R-CNN detector to complement the accuracy for small pedestrian detection [16].

The major challenge faced was the system's ability to identify vehicles' accidents. Accidents result in deformity of more than one vehicles when in contact. While the non-accident database may contain any image other than accident, such as, pedestrians, birds, and trees, there are attributes unique to every accident image. This problem of object detection and analysis being further essential has been met by making the accident dataset more concentrated than the non-accident dataset.

The major contributions of this paper are:

1. A CNN collision detection system that delivers 100% accuracy with full score in precision, recall and f1-score, which is lossless in nature.
2. A low computation and run time system feasible for real world applications.
3. Result and analysis of the different config. of 108 CNNs and their trends.

The rest of this paper is organised as follows - Sec. 2 presents the preliminaries, Sec. 3 provides information on the database, Sec. 4 describes the PM, Sec. 5 includes information on methodologies, Sec. 6 presents the analysis' results and finally, conclusion is mentioned in Sec. 7.

## 2 Preliminaries

**Gaussian Mixture Model (GMM)** [15] is used for better feature extraction.

**Kaiming weights, and biases (W, B)** [11] work efficiently with non-linear activations, especially ReLU, to help the model converge much easily.

**Dropout on 1$^{\text{st}}$ fully connected layer and a 2$^{\text{nd}}$ fully connected layer (DFc)** does not let extreme or rare data values affect model's results by being biased.

**Average Pooling (AvgPool)** helps in detection of smoother features, in comparison to **MaxPool** that detects edges and corners more efficiently.

**Padding = 0 on 1$^{\text{st}}$ convolution layer (C1P(0))** reduces image dimensions and saves memory .

**Negative Log Likelihood Loss (NLLLoss)** function considers the uncertainty of the prediction based on deviation from the actual class, represented as:

$$L(y) = -log(y) \tag{1}$$

**Multi-class Hinge Loss (Hinge)** is widely used for classifications with SVMs.

**Cross Entropy** loss function is the average difference between the true and predicted probability distributions which minimizes nearing to zero (ideal score).

**Stochastic Gradient Descent (SGD)** optimizer is cheaper, noisier and takes more steps to reach the minima, unlike Gradient Descent. SGD is used with either no momentum (**SGD(0)**) or with momentum = 0.9 (**SGD(0.9)**).

**Adam** optimizer is a first-order gradient that requires little memory. Individual adaptive rates are calculated for varying parameters from $1^{st}$ and $2^{nd}$ gradient moments' estimates[12].

## 3 Database used

While certain laws prohibit data retrieval for detecting vehicle collisions, minimal to no data is available on open data platforms as accidents are rare events [5] and adequate amount of cameras are not being used on devices and roads[17]. Dataset for this work has been collected via web scraping from Google and Accident Images Analysis Dataset [14].

## 4 The proposed model

In the PM as visualized in Fig. 1, image processing techniques have been performed on input images, which are changed into grayscale color composition and transformed into tensors of constant dimensions (28 x 28 x 1). Grayscale aids in feature extraction while consuming less memory with 1 color channel as compared to 3 color channels of RGB. The analysis has been performed on 8,284 datapoints.

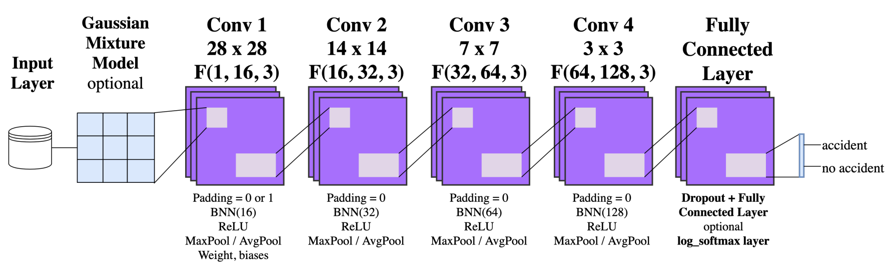


**Fig. 1** The proposed CNN with config. to classify the images as 'with accident' or 'without accident'. Denotions used are *Conv*: Convolution Layer, *BNN*: 2D Batch Normalization, *ReLU*

Before our system is subjected to classification, some of the config. involve using GMM on the input tensors before being fed to the PM. The PMs are fit to the needs with the below described config., and trained on 30 epoch iterations with the train loader divided into batches of 20.

- This CNN comprises of 4 convolution layers with a 3x3 kernel and padding = 1 each. First convolution layer of some config. has no padding.
- The output channels of the convolution layers which are 16, 32, 64, 128 respectively undergo 2D batch normalization, ReLU activation and then pooling.
- Pooling is either MaxPool or AvgPool, of kernel size 2x2 and stride 2.
- The CNN has one fully connected layer and a log soft max layer to convert all of the final output channels into a distribution of class scores.
- Some config. of the CNN adds DFc before the log soft max layer.

## 5 The process

The dataset is split into a 4:1 ratio by keeping 80% of the data as a training set, split into batches of 20 and 20% of data as a testing set, split into batches of 22.

In Fig. 2, the test performance of CNN 1 of Table 6.2 is visualized on a batch of test images. The classifier's aim is to detect accidents in images. A "True" label signifies that an accident has taken place, whereas a "False" label signifies otherwise.

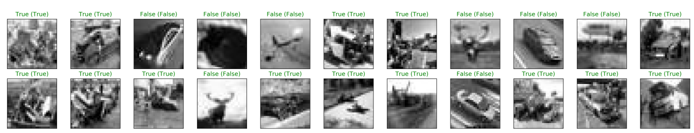


**Fig. 2** Test performance of CNN 1 of Table 6.2: W, B + Hinge loss + Adam's optimizer. Every image is captioned in the format of *Predicted Class (Actual Class)*

The Table 6.1 depicts the baseline CNNs created on which further config.urations were made with addition and removal of: GMM, [W, B], DFc, [AvgPool or MaxPool], C1P(0), [NLLLoss, Cross Entropy, Hinge] loss, [SGD(0), (SGD(0.9), Adam] optimizer. These resulting 108 CNNs were categorized into- the best performing model (illustrated in Table 6.2) and the worst performing models (illustrated in Table 3) based on the following parameters:

**Accuracy, Precision, Recall, F1 Score** [19] The PM has results with a full score in classification report (CR) which signifies an accuracy of 100% with precision, recall and f1-score to be 1.0/1.0 in score, the ideal score.

**Loss** of 0.00 is the ideal score for the system which the PM has achieved.

**Training time, Test time** are more preferable with less values as they help ensure the system is more robust and capable for immediate action.

## 6 Results and Discussion

The presented computation were performed using a system with 16GB RAM, a core i7 process and an Intel series GPU, all operation were primarily performed on the terminal using python scripting language.

### 6.1 *The Base Models*

Table 6.1 as shown below illustrates the significant baseline models created for this analysis, which are further configured resulting into 108 permutations of CNNs.

**Table 1** The base models

| | CNN Configuration | Loss function | Optimize function | Accuracy 0 | Accuracy 1 | Accuracy Overall | Precision 0 | Precision 1 | Recall 0 | Recall 1 | F1-score 0 | F1-score 1 | Loss | Training Time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | W, B + MaxPool | Hinge | SGD(0) | 99 | 99 | 99.71 | 1.0 | 1.0 | 1.0 | 0.99 | 1.0 | 1.0 | 0.006895 | 2:29.33 |
| 2 | W, B + MaxPool | Cross Entropy | SGD(0) | 99 | 99 | 99.67 | 1.0 | 0.99 | 1.0 | 0.99 | 1.0 | 0.99 | 0.006895 | 2:15.35 |
| 3 | MaxPool | Cross Entropy | SGD(0) | 92 | 96 | 93.92 | 0.98 | 0.85 | 0.93 | 0.96 | 0.96 | 0.90 | 0.170758 | 3:40.21 |

These are:
- The CNN 3 has Cross Entropy loss function and SGD(0) optimizer.
- The CNN 2 improves drastically by 5.75% on addition of W, B.
- The CNN 1 sees a further improvement with the use of Hinge loss function.

### 6.2 *Best Performing CNNs*

- In Table 6.2, CNNs with W, B + MaxPool + Hinge loss perform well.
  - CNN 1, 2 and 3: *0.00 loss* is achieved with 3 of the 7 CNNs which give *full score in CR*. The CNNs with run time lower than these 3 CNNs have a negative effect on their performance. Hence, CNN 1 with Adam runs in the *fastest testing time* of 108 CNNs, given it performs well.
  - CNN 6: gives loss = 0.00014 with SGD(0.9).
  - CNN 4 with DFc gives loss of 0.000119, CNN 5 is similar with +GMM.
  - CNN 7: with AvgPool + C1P(0), the loss decreases to a slight, 0.000297.

**Table 2** The Best Performing Models

| | CNN Configuration | Loss function | Optimize function | Accuracy 0 | Accuracy 1 | Accuracy Overall | Precision 0 | Precision 1 | Recall 0 | Recall 1 | F1-score 0 | F1-score 1 | Loss | Training Time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | W, B + MaxPool | Hinge | Adam | 100 | 100 | 100 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0 | 2:31.95 |
| 2 | GMM + W, B + MaxPool | Hinge | SGD(0.9) | 100 | 100 | 100 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0 | 2:25.48 |
| 3 | GMM + W, B + MaxPool | Hinge | Adam | 100 | 100 | 100 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0 | 2:33.49 |
| 4 | W, B + MaxPool + DFc | Hinge | SGD(0.9) | 100 | 100 | 100 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.000119 | 3:50.20 |
| 5 | GMM + W, B + MaxPool + DFc | Hinge | SGD(0.9) | 100 | 100 | 100 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.000119 | 3:51.20 |
| 6 | W, B + MaxPool | Hinge | SGD(0.9) | 100 | 100 | 100 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.00014 | 2:31.76 |
| 7 | W, B + C1P(0) + AvgPool | Hinge | SGD(0.9) | 100 | 100 | 100 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.000297 | 2:34.28 |
| 8 | GMM + W,B + MaxPool | Cross Entropy | SGD(0.9) | 100 | 100 | 100 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.000346 | 2:16.82 |
| 9 | AvgPool | Cross Entropy | SGD(0.9) | 100 | 99 | 99.99 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0019 | 2:34.80 |
| 10 | GMM + W, B + MaxPool | Hinge | SGD | 99 | 100 | 99.95 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 0.0012 | 2:34.14 |

- CNN 9: With AvgPool + Cross Entropy + SGD(0.9), yields a 1.0 recall, precision and f1 score and 99.99% of accuracy with 99% accuracy of the accident data and 100% accuracy of the non-accident data.
- GMM + W, B + Hinge loss has resulted in one of the best performing CNNs.
  - Only CNN 10, with SGD(0) has accuracy of non-accident dataset greater than accuracy of accident dataset even though it is smaller in size.

  - CNN 2 with SGD(0.9) and CNN 3 with Adams yield *full score in CR and 0.00 loss*, and differ in training and testing time.
  - CNN 8: With Cross Entropy and SGD(0.9) gives *full score in CR*, loss = 0.000346.
- CNNs with NLLLoss have not made it to the well performing CNNs.

### *6.3 Worst Performing CNNs*

- In Table 3, CNN 13: With W, B + C1P(0)+ AvgPool + Cross Entropy + SGD(0.9) performs considerably well with an overall accuracy of 99.79%, a CR tending to 1.0, and a loss=0.009001. Yet, its second highest training time (7:24 minutes) attribute makes it amongst the worst performing CNNs.
- NLLLoss + SGD(0) is the most common duo to yield CNNs with one of the poorest performances when paired with W, B + DFc + MaxPool (CNN 11), AvgPool (CNN 10), GMM + MaxPool (CNN 2) and MaxPool (CNN 1).:
  - CNN 10: With AvgPool, runs with the third slowest training speed (6:15 minutes) and has the slowest testing speed (5.1658 seconds).
  - CNN 1: With MaxPool, performs amongst the worst with 100% accuracy of the accident data and 0% accuracy of the non-accident data.

**Table 3** The Worst Performing Models

| | CNN Configuration | Loss function | Optimize function | Accuracy | | | Precision | | Recall | | F1-score | | Loss | Training Time |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 0 | 1 | Overall | 0 | 1 | 0 | 1 | 0 | 1 | | |
| 1 | MaxPool | NLLLoss | SGD(0) | 100 | 0 | 70.63 | 0.71 | 0.00 | 1.0 | 0.00 | 0.83 | 0.00 | 0.580795 | 0:06.50 |
| 2 | GMM + MaxPool | NLLLoss | SGD(0) | 98 | 5 | 70.71 | 0.71 | 0.63 | 0.99 | 0.05 | 0.83 | 1.00 | 0.006895 | 0:02.78 |
| 3 | AvgPool | Hinge | SGD(0) | 95 | 60 | 84.75 | 0.85 | 0.85 | 0.95 | 0.6 | 0.9 | 0.71 | 0.3496 | 2:29.05 |
| 4 | MaxPool | NLLLoss | Adam | 79 | 98 | 85.07 | 0.99 | 0.67 | 0.79 | 0.99 | 0.88 | 0.8 | 0.672155 | 3:22.70 |
| 5 | GMM + AvgPool | Hinge | SGD(0) | 90 | 82 | 88.49 | 0.93 | 0.79 | 0.91 | 0.83 | 0.92 | 0.81 | 0.2996 | 2:31.56 |
| 6 | AvgPool | Hinge | SGD(0.9) | 89 | 88 | 89.21 | 0.95 | 0.78 | 0.9 | 0.88 | 0.92 | 0.83 | 0.2637 | 2:32.20 |
| 7 | MaxPool | Hinge | SGD(0) | 99 | 65 | 89.5 | 0.87 | 0.99 | 1.0 | 0.66 | 0.93 | 0.79 | 0.266 | 3:35.20 |
| 8 | W, B + MaxPool + DFc | Hinge | Adam | 99 | 75 | 92.29 | 0.91 | 1.0 | 1.0 | 0.75 | 0.95 | 0.86 | 0.2777 | 9:04.75 |
| 9 | AvgPool | Hinge | Adam | 99 | 77 | 92.63 | 0.91 | 0.98 | 0.99 | 0.77 | 0.95 | 0.86 | 0.3754 | 2:43.98 |
| 10 | AvgPool | NLLLoss | SGD(0) | 93 | 94 | 94.16 | 0.98 | 0.86 | 0.94 | 0.95 | 0.96 | 0.9 | 0.1677 | 6:15.65 |
| 11 | W, B + MaxPool + DFc | NLLLoss | SGD(0) | 95 | 91 | 94.6 | 0.97 | 0.9 | 0.96 | 0.92 | 0.96 | 0.91 | 0.16909 | 3:56.25 |
| 12 | W, B + MaxPool + DFc | NLLLoss | Adam | 98 | 89 | 96.13 | 0.96 | 0.97 | 0.99 | 0.9 | 0.97 | 0.94 | 0.3038 | 3:44.28 |
| 13 | W, B + C1P(0) + AvgPool | Cross Entropy | SGD(0.9) | 100 | 99 | 99.79 | 1.0 | 1.0 | 1.0 | 0.99 | 1.0 | 1.0 | 0.009001 | 7:24.04 |

- The CNN with W, B + DFc yielded 3 of the worst performing CNNs:
  - With NLLLoss + Adam, CNN 12 performs better than CNN 3 for using MaxPool and not AvgPool.
  - CNN 8: With Hinge + Adam, trains in the slowest time (over 9 seconds).
- Unlike CNN 9 of Table 6.2, CNNs using AvgPool over MaxPool have not performed well when used with Hinge + SGD(0) (CNN 3), Hinge + SGD(0.9) (CNN 6), Hinge + Adam (CNN 9) and NLLLoss + SGD(0.9) (CNN 10).
- Like the CNN 10 of Table 6.2, CNN 10 with AvgPool + NLLLoss + SGD(0.9) and CNN 4 with NLLLoss + Adam of Table 3 have yielded a higher accuracy for non-accident dataset than the accident dataset given that the dataset of accidents was much larger than that without accident images.

### *6.4 Trends*

- NLLLoss has considerably affected the CR and loss *negatively*.
- CNNs with W, B have significantly better performance than baseline CNNs.
- GMM + W, B tends to *enhance the CNN's performance further*.
- Hinge loss function improved the CNN's performance, especially when complemented with SGD(0.9).
- SGD(0.9) has shown *better results* as compared to SGD(0).
- DFc increased the training time for most CNNs, hence its effect is subjective.

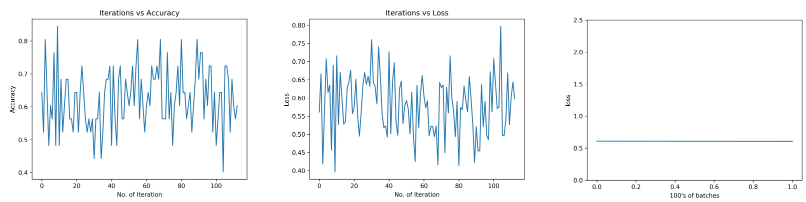


**Fig. 3** Worst performing network - Table 3 Network 1 uses NLLLoss

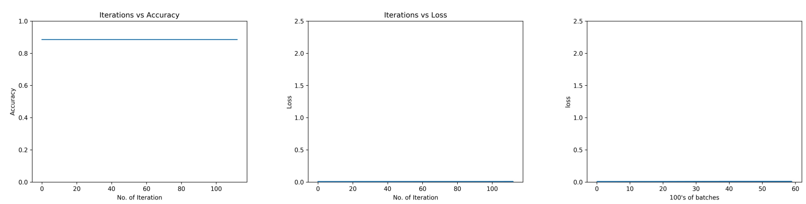


**Fig. 4** Best performing network - Table 6.2 Network 1 uses Hinge loss

Figure 3 and 4 show the changes in 3 graphs: The first one determines the number of iterations on which the model is trained vs. accuracy, second depicts iteration vs. loss, and the third depicts changes in loss with every 100 batch.

## 7 Conclusion

In autonomous driving, detecting vehicle collisions minimizes the risk of accidents in real world implementable scenarios. This paper proposes a collision detection system by forming a baseline CNN on which further models are derived with different configurations resulting in analysis of 108 CNNs for the possible methods to use for vehicle accident classification on collision image dataset such that there is maximum feature extraction. Additionally, the proposed models yield full score in CR. Hence, the PMs are efficient, highly accurate and lossless in nature. The results have been achieved in minimal computation cost, that is, by consuming less memory and time making the entire system feasible for application in real world scenarios.

## References

1. Petrović D., Mijailović R. and Pešić D.: Traffic Accidents with Autonomous Vehicles: Type of Collisions, Manoeuvres and Errors of Conventional Vehicles' Drivers, Transportation Research Procedia, vol.45, pp. 161-168. (2020)
2. Combs S., Sandt S., Clamann P., Mcdonald C.: Automated Vehicles and Pedestrian Safety: Exploring the Promise and Limits of Pedestrian Detection. Am. J. Prev. Med. 56, 1–7. (2019)
3. Ahonen, T., Hadid, A., and Pietikainen, M.: Face description with local binary patterns: Application to face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, **28**(12), 2037–2041 (2006)
4. Cao, X., Lan, J., Yan, P., and Li, X.: Vehicle detection and tracking in airborne videos by multi-motion layer analysis. Machine Vision and Applications, **23**(5), 921–935 (2012)
5. Chan FH., Chen YT., Xiang Y., Sun M. (2017) Anticipating Accidents in Dashcam Videos. In: Lai SH., Lepetit V., Nishino K., Sato Y. (eds) Computer Vision – ACCV 2016. ACCV 2016. Lecture Notes in Computer Science, vol 10114. pp. 136–153. Springer, Cham (2017).
6. Chen H., Tsai SS., Schroth G., Chen DM., Grzeszczuk R. and Girod B.: Robust text detection in natural images with edge-enhanced Maximally Stable Extremal Regions, In: 2011 18th IEEE International Conference on Image Processing, pp.2609-2612, IEEE, Brussels (2011)
7. Bay H., Tuytelaars T., Van Gool L.: SURF: Speeded Up Robust Features. In: Leonardis A., Bischof H., Pinz A. (eds) Computer Vision – ECCV 2006. ECCV 2006. Lecture Notes in Computer Science, vol 3951. Springer, Berlin, Heidelberg (2006)
8. Dogru N. and Subasi A.: Traffic accident detection using random forest classifier. In: 2018 15th Learning and Technology Conference (LT), 40-45. IEEE, Jeddah (2018)
9. Guo, J., Kurup, U., and Shah, M.: Is it safe to drive? an overview of factors, challenges,and datasets for driveability assessment in autonomous driving. ArXiv, abs/1811.11277 (2018).
10. He K., Zhang X., Ren S. and Sun J.: Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778. IEEE, Las Vegas, NV (2016)
11. He K., Zhang X., Ren S. and Sun J.: Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In: 2015 IEEE International Conference on Computer Vision (ICCV) (ICCV '15), pp. 1026–1034. IEEE Computer Society, USA (2015)
12. Kingma DP. and Ba J.: Adam: A Method for Stochastic Optimization. In: International Conference on Learning Representations. (2014)
13. Dogru N. and Subasi A.: Traffic accident detection by using machine learning methods. (2012)
14. Pashaei, A., Ghatee, M. and Sajedi, H. Convolution neural network joint with mixture of extreme learning machines for feature extraction and classification of accident images. J Real-Time Image Processing. IEEE. (2019).
15. Reynolds, DA. Gaussian mixture models. In: Encyclopedia of Biometrics. Springer, Boston, MA (2009)
16. Shah, A.P., Lamare, J., Nguyen-Anh, T., and Hauptmann, A.G.: CADP: A Novel Dataset for CCTV Traffic Camera based Accident Analysis. In: 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp.1-9. Auckland (2018)
17. Shah, A.P., Lamare, J., Nguyen-Anh, T., and Hauptmann, A.G.: Accident Forecasting in CCTV Traffic Camera Videos. ArXiv, abs/1809.05782. (2018)
18. Wang X., Han TX. and Yan S.: An HOG-LBP human detector with partial occlusion handling. In: 2009 IEEE 12th International Conference on Computer Vision, pp. 32-39. Kyoto, (2009)
19. Tharwat, A.: Classification assessment methods. Applied Computing and Informatics. (2018)