International Journal of Scientific Research and Engineering Development-– Volume 8 Issue 2, Mar-Apr 2025
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

Vehicle Accident Detection System using Deep Learning Model

Kakade Ravindra[1], Wagh Sanika [2], Sonawane Tejaswi[3], Telore Samruddhi[4], Bhambare Pranjal[5], Sonar Suyog[6]

*1 HOD. Department of Computer Technology, Padmashri Dr. Vitthalrao Vikhe Patil Institute of Technology & Engineering(Polytechnic), Loni, Maharashtra, India

*2,3,4,5 Department of Computer Technology, Padmashri Dr. Vitthalrao Vikhe Patil Institute of Technology & Engineering(Polytechnic), Loni, Maharashtra, India

----------**********----------

Abstract:

For the time being, car accidents are the leading cause of fatalities. There appears to be a correlation between the increase in vehicle deaths and the growth in automobile manufacturing. Due to an increasing trend of people being careless, which has serious implications, metropolitan regions also show a higher mor tallit rate. Many people believe that the increased frequency of car accidents is due in large part to the fact that there are so many different modes of transportation available nowadays. The increasing number of cars using the road networks is causing the trend in traffic, both vehicle and passenger, to increase. They pose a risk to the nervous and physiological systems of the body. Consequently, automatic accident detection methods that are both practical and based on principles from the field of image processing are required. Image normalization methods, Mobile Net Neural Networks, and a Decision Tree algorithm for auto accident detection are all part of the methodology used to accomplish this goal.

Keywords: Mobile Net Neural Networks, car accident detection, Image processing.

----------**********----------

I. INTRODUCTION

Along with the rising demand for and widespread use of automobiles, the number of traffic accidents and fatalities has also risen. The number of people killed in car accidents is higher now than it has ever been. The monetary toll that drivers, their families, and countries pay as a result of car accidents is staggering. Delays in aid have tragically led to the deaths of many people. Failure to offer first aid promptly following an injury, no matter how minor, could have devastating consequences. The likelihood of the driver or passengers being able to call for assistance in an emergency is high. Avoiding unintentional deaths is of the utmost importance. Even while accidents can't be stopped, scientists can make sure injured people get medical help quickly. Every second counts for rescue workers in a situation like this. The delayed arrival of the ambulance causes multiple fatalities. The most likely explanation for the time gap is that nobody witnessed what happened. Automobile accidents cause a disproportionate number of fatalities and major injuries. Deaths due to inadequate emergency care outweigh those due to car accidents by a significant margin. Despite the unpredictability of most car accidents, many are preventable with reliable and quick emergency medical treatment. Accidents can happen for a variety of reasons, including mechanical failure, bad traffic conditions, and drivers who are slow to react. There is a significant risk of death for the sufferer if they do not receive prompt medical treatment after an incident that causes harm. Delays in notifying emergency services are the main cause of the increased duration of ambulance turnaround times. Unfortunately, when people are unable to do

simple tasks, their capacity to could jeopardize the ability to start an emergency call.

The general public has to start paying more attention and learning more about this issue. Those without access to vehicles who must navigate the city on foot face a comparable risk. Minute after minute, the situation is becoming worse. As a result of its pervasiveness in people's daily lives, drive-by traffic affects many different products and pastimes. There are a lot of individuals hurt or killed in car accidents every year, and even more others who sustain injuries or become disabled.

[1] The programmed accident detection system presented by Dr. C. K. Gomathy et al. has the potential to save lives of those who have been in accidents. The suggested system is so intuitive that even someone without technical training should have no trouble learning how to use it. Hardware and software components make up the system. Installed in the car, the equipment unit has sensors for accident detection that are controlled by an Arduino board. However, the code is an Android software that drivers install on their smartphones; this app is what actually gets the drivers a point-by-point map. [2] in The authors V. Veera Anusuy et al. detail a new method for detecting accidents. Identifying and reporting accidents is the focus of the proposed solution. Then, after reading it, it notifies the nearest emergency service provider of the precise location of the struck vehicles. Various system devices can be communicated with with the help of Arduino.

[3] the third In this section, Duaa Hadi Nassar et al. summarize the system's results and provide a brief overview of its underlying principles and goals, which include the detection of road traffic accidents and the transmission of notification messages to specialists in order to address the issues that arise as a result of these accidents. Machine learning, and more specifically deep learning methods in accordance with the supervised deep learning approach, formed the basis of the system's construction. The datasets have been classified using a CNN-SVM hybrid model, which combines the two methods into a single model by incorporating the SVM algorithm within the CNN algorithm as a layer.

II. LITERATURE SURVEY

[4] According to Dr. M. Sarojini Devi et al. numerous individuals have tragically lost their lives in our nation due to accidents, causalities, or poor communication. This led to the installation of a system that can detect theft and accidents in vehicles automatically. With this technique, the author can respond quickly to accidents, reducing the number of fatalities. In addition, it has a role in detecting thefts. In this project, we utilize a system that can recognize faces and detect when people blink their eyes.
[5] Heesoo Kim and colleagues presented Despite the growing number of AV accidents, investigations into these incidents are nevertheless carried out at the same level as those involving conventional vehicles. There is an immediate need for a system that can determine the root cause of accidents involving autonomous vehicles and other high-tech vehicles, since the number of these vehicles is expected to grow in the coming years. The research yields things that can be derived from several angles and puts into action a system that can readily understand the intricate relationships between them.

[6] According to Andy Berres et al., this publication introduces an annotated accident dataset that combines data on traffic accidents with data on weather, light, and radar detection sensors. The dataset is presented in a way that makes it easy to process using databases, data workflows, or machine learning tools.[7] The narrate system by T V Pavan Kumar et al., which is on point number seven, has a layout that is small, lightweight, and cheap. Its vibration sensor, GPS, and Internet of Things connectivity all work together to make it safer. It also fixes a lot of problems with automated methods that try to find accidents. Consequently, many lives are saved since the location may be found more quickly, allowing for immediate treatment.

[8] According to Shehzad Aslam et al., a growing number of individuals are hurt or killed in automobile accidents. To overcome these obstacles, this research proposes a novel Internet of Things (IoT)-based automated vehicle accident detection and visual status reporting system. By utilizing in-vehicle sensors, the system is able to identify accidents independently. [9] The hardware components that Sadda Bharath Reddy et al. outlined are essential for an accident detection and alert system that is based on Android. All of the system's sensors, data processing, and control are handled by a low-power microprocessor. For the detecting algorithms and software to function, there must be enough memory and storage. A sturdy, automotive-grade housing that can endure the vibration, stress, and temperature extremes experienced within a car is required for all electronics. In the event that the vehicle's electrical system experiences damage, a backup battery will provide power. Accidents can be detected and the location of the occurrence can be automatically notified to emergency personnel by using the system that is built into Android devices and their wireless connectivity.

[10] A hybrid deep learning approach integrating LSTM and GBRT models is presented by Tanusree Chatterjee et al., who aim to build an automated system for detecting and analyzing the severity of automotive accidents. Because of its effectiveness with time-variant data, or datasets with temporal features, LSTM has been selected for this project. Since GBRT employs ensemble learning, it has been added to the dataset after LSTM in an effort to boost the models' performance.

[11] the eleventh In order to identify the noises of vehicle crashes on roads, Nur Nazifah Adlina Mohd Nazeem et al. detail a camera controller based on microcontrollers that use TDOA. By automatically directing the camera to record the sound of a vehicle crash, this technology can detect accidents on highways and allow authorities to respond quickly to save lives, all while minimizing the need for human resources to constantly patrol the roads.

[12] To tackle the difficulty of creating trac0c accident pictures and inferring accident links, Chun-Hao Wang et al. introduce a deep learning strategy driven by multisource accident datasets. Through the use of deep learning approaches, knowledge graph-based portraiture and reasoning can be implemented, and models and algorithms can be proposed for the construction of multisource accident data from semi-structured and unstructured data. In order to help with real-life police traffic accident prevention situations, we are now working on adding more different data types to the dataset and using the models and outcomes that have been generated.

[13] According to Deeksha K et al., one step toward making road safety measures more effective is the application of deep learning algorithms for accident severity prediction. By integrating convolutional neural networks (CNNs) and long short-term memory (LSTM) networks, the approach reliably predicts if an occurrence is moderate, severe, or neither. Researchers clearly put a lot of time and effort into this study because of the detailed methodology, which includes data pre-processing employing ensemble methods during the modeling stage, as well as appropriate model training and evaluation.

[14] Victor A. Adewopo et al.'s 14] goal is to find the most advanced action recognition system for smart city autonomous transportation and accident detection. Based on the inclusion and exclusion criteria given in the "Literature search" section, the author followed the PRISMA standard to identify seminary articles pertaining to our issue domain. From a starting list of 2030 articles, the author chose 21 papers to study. Then, using the three pillars of our research question, the author categorized and rated the papers. In this article, we looked at the most important methods and uses of action recognition for driverless cars.

[15] Combining K-nearest neighbor with Pearson correlation variables can enhance deep learning models' object identification capabilities, according to Shubham gade et al. The author To disperse dynamic power among the charging ports in electric

vehicle charging stations, a thorough assessment of the dataset is performed to preprocess the attributes according to their requirements. During the preprocessing step, the author use the Pearson correlation matrix to eliminate superfluous columns and establish correlations. Since the author's proposed system is engineered to reroute power to the charging connection, a complete recharge of the vehicle's batteries is one potential result. To determine which fast charging port has the least amount of remaining battery life, the created model employs the K-Nearest neighbor technique. Using the data-based model of the deep belief system. This method, which is based on K-nearest neighbors, will help discover the accidents that are visually similar.

.

III PROPOSED METHODOLOGY

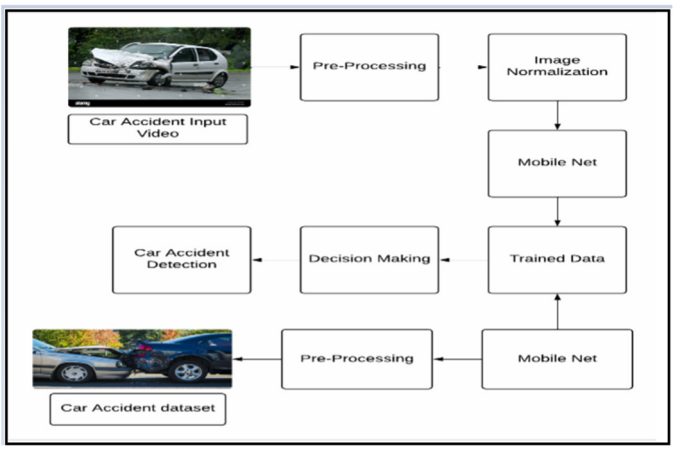

Fig 1: System Overview Diagram

The method that has been suggested to establish a Vehicle accident detection system depicted in the system overview in Figure 1 up top. The suggested method was based in part on the execution of the procedures detailed below.

Step 1: Dataset Generator : To build an interactive graphical user interface, developers use the NetBeans integrated development environment (IDE) and the Java programming language's Swings Framework. Supplying the system with the CCTV footage is the objective of this user interface. Following accurate decipherment of the CCTV footage, the required byte data is generated and delivered to proceed with the Blockchain construction process.

Step 2: Data Labelling : The previous stage's recorded photographs are utilized to classify them using the labeling software in this step. To label an image, the user must load it into the labeling program and provide the coordinates of the top right corner (x1) and the bottom left (x2). the rectangle's y-axis, and x2, y2-for the coordinates in the bottom right corner. In order to train the model using a deep learning network called MobileNet, the acquired coordinates are saved in an.xml file.

Step 3: Installation of APIs: To begin, on this Google Colab instance, we need to install the TensorFlow Object Detection API. A handful of installation scripts and a clone of the [TensorFlow models repository (https://github.com/tensorflow/models) are needed for this. The following code portions can be run by clicking the play button. The next step in downloading and extracting cuDNN files is to install the conda environment. The next thing to do is to copy the tensorflow models repository off of GitHub. After that, the Object Detection API has to be installed.

Step 4: Upload Image Dataset and Prepare Training Data: Here, we'll get the TensorFlow training data ready by uploading our training photos and running the scripts for TF Record generation. To begin making TF Records from our data, we need to upload our photos, divide them into a train, validation, and test folder, and then execute scripts. To begin with, regarding create a single folder named "images.zip" on our local PC and compress all of the training images and XML files within. All of the files have to be contained within the zip folder. After it has been uploaded, we need to execute some commands in order to extract the zip file and configure our picture folders. The /content folder is where these folders are created in this instance's file system. To access the file system, we can use the "Files" icon located on the left side of the screen.

Step 5: Split images into train, validation, and test folders: Here you can see our "images.zip" file among the listed files, and you can also see the folder icon on the left. After the dataset has been uploaded, the next step is to extract its contents and organize them into folders for the images. In this particular instance, the /content folder is where these folders are generated. It is possible for us to peruse go to the file system, just click the "Files" icon over there on the left. The next step is to divide the photos into three sets: train, validation, and test. The purpose of each set is as follows: In order to train the model, these photos were really used. At each stage of training, the neural network is fed a new batch of photos from the "train" set. In

these pictures, the network identifies what things are and where they are located. By using back propagation, the training algorithm determines the loss and modifies the network weights. For validation purposes, the training algorithm can use images from the "validation" set to evaluate how well it is doing and to tweak hyperparameters (such as the learning rate). Training with these images is more frequent than with "train" images; in fact, it's more frequent than once every a predetermined number of steps. Warning: The neural network will never view these pictures while it is being trained. A human being should use them to do the last tests on the model to ensure its accuracy.

Step 6: Create TF Records: Last but not least, we must transform the photos into TF Records, a data file format utilized by TensorFlow for training purposes. To automate the process of converting the data to TF Record format, we are utilizing Python scripts. We must first establish a class label map before we can run them. The The code snippet below will generate a "labelmap.txt" file with a catalog of classes. Simply insert a new line for each of our classes (such as "car image" and "car accident image") in place of the existing "class1" and "class2" content. After that, press play to run the code. As a result, the object detection model's detectable classes are recorded in a "labelmap.txt" file.

Step 7: Set Up Training Configuration : Here, we configure the training environment for an SSD-MobileNet model. Specifically, we're choose which of the TensorFlow 1 Object Detection Model's pre-trained models to employ. There is a configuration file included with each model that specifies where to find certain files. determines a variety of training parameters, including the learning rate and the overall number of steps—and more. We are making changes to the custom training job's configuration file. Some models that are available in the TF1 Model Zoo are listed in the first piece of code, which also defines some filenames that will be used to download the model and configuration file later on. Because of this, keeping track of the models we're using and adding more models in the future is a breeze. To train with a specific model, set the "chosen_model" variable to its name. At the moment, the "ssd-mobilenet-v1-quantized" model is selected to be used. After that, specify and download the settings and pre-trained model files by clicking play in the next three parts. We need to add some general training parameters to the configuration file after downloading the model and configuration file. To regulate the various stages of training, we employ the following variables.

num steps: The sum of all steps to be used when training the model. For an initial target, 40,000 steps should be sufficient. If, once training is complete, we see that the loss measures are still going down, we can add extra stages. It takes more time to train if there are more steps. Additionally, training might be halted. prematurely if the loss levelizes before the prescribed number of steps is reached.
batch_size: The picture count for each training run. The amount of accessible GPU memory for training determines the maximum batch size, which in turn reduces the number of steps required to train a model. It is generally recommended to use 16 GPUs for Colab instances.Quantization-aware training is the only one that uses quant_delay_steps. Following these extensive procedures, the training algorithm

will simulate quantization by inserting "fake" quantization nodes into the network. The half-way point of the entire training sequence is a decent place to begin. Visit [thislink](https://neuralet.com/article/quantization-of-tensorflow-object-detection-api-models/) for further details.During this step, other training information is also assigned, such as the location of the pre-trained model file, the config file, and the total number of classes. In order to apply the training parameters that we have just defined, we will need to edit the configuration file. In order to create our own "pipeline file.config" file, the following code will automatically replace the required parameters in the.config file that was downloaded. Include our dataset, model checkpoint, and training parameters in the main pipeline file to create a custom configuration file. The following block alters the script for training to save checkpoints at 1000 steps intervals. Whether we want to store checkpoints more or less frequently, we can adjust 'num_eval_steps' accordingly. In table 1 below, you can observe the mobile-network architecture.

Type	Filter Shape	Input Size
Conv1	3×3×3×32	224×224×3
Conv2 dw	3×3×32 dw	112×112×32
Conv2 pw	1×1×32×64	112×112×32
Conv3 dw	3×3×64 dw	112×112×64
Conv3 pw	1×1×64×128	56×56×64
Conv4 dw	3×3×128 dw	56×56×128
Conv4 pw	1×1×128×128	56×56×128
Conv5 dw	3×3×128 dw	56×56×128
Conv5 pw	1×1×128×256	28×28×128
Conv6 dw	3×3×256 dw	28×28×256
Conv6 pw	1×1×256×256	28×28×256
Conv7 dw	3×3×256 dw	28×28×256
Conv7 pw	1×1×256×512	14×14×256
Conv8 dw	3×3×512 dw	14×14×512
Conv8 pw	1×1×512×512	14×14×512
Conv9 dw	3×3×512 dw	14×14×512
Conv9 pw	1×1×512×512	14×14×512
Conv10 dw	3×3×512 dw	14×14×512
Conv10 pw	1×1×512×512	14×14×512
Conv11 dw	3×3×512 dw	14×14×512
Conv11 pw	1×1×512×512	14×14×512
Conv12 dw	3×3×512 dw	14×14×512
Conv12 pw	1×1×512×512	14×14×512
Conv13 dw	3×3×512 dw	14×14×512
Conv13 pw	1×1×512×1024	7×7×512
Conv14 dw	3×3×1024 dw	7×7×1024
Conv14 pw	1×1×1024×1024	7×7×1024
Avg Pool	Pool 7×7	7×7×1024
FC	1024×25	1×1×1024
Softmax	Classifier	1×1×25

Fig 2: MobileNet Body Architecture

Step 8: Train Custom TF1 Object Detector: Our object detection model needs training here! The TF Object Detection API's "model_main_tf2.py" script is used to train the model. All of the arguments and parameters utilized by "model_main_tf2.py" in earlier parts of this collaborate document. Time required for training can range from two to six hours, with exact timing dependent on model, batch size, and training procedures. As a result, we will have a.tflite file to use for model testing.

Step 9: Testing the model for car accident detection : At this stage, the Python program imports the Android Cam app, which is compatible with both mobile phones and laptops, and uses the phone's camera to record video and, in turn, the frames. The file that contains the learned model. A tflite detects the vehicle crash in the frames of the live broadcast. Authorities are notified of the exact location of the vehicle accident as soon as it is detected.

IV RESULTS AND DISCUSSIONS

On a Windows workstation, the Python programming language is used to implement the proposed methodology for identifying car accidents. To implement the aforementioned plan into code, we can use the Spyder IDE. The computer used for the rollout has a 1 terabyte hard drive, 8 GB of random access memory, and an Intel Core i5 CPU.

Performance Evaluation through Root Mean Square Error

By calculating the root mean square error (RMSE), we may find the error rate of the proposed method. The experiment uses the Root Mean Square Error (RMSE) to measure the difference between the expected and observed results of the Mobile Net module's car accident detection. One can see the RMSE approach in Equation 1.

$$\mathrm{RMSE}_{fo} = [\sum_{i=1}^{N} (z_{f_i} - z_{o_i})^2 / N]^{1/2}$$

Where

$\sum$ - Summation.

$(Z_{fi} - Z_{oi})^2$ - Differences Squared for the Car accident Detection.

N - Number of Images.

The Mean Square Error (MSE) must be computed before the Strategy Error Rate (ERR) can be estimated using Root Mean Square (RMSE). The

MSE is a measure of the difference between the expected and observed values of vehicle accident detection. The project is currently being tested with an increasing number of trials, and the results are being recorded in Table 1. Below, you can see Figure 3, which is a diagram generated from the obtained outcomes.

S no.	Number of Iterations	Correctly identified Car accidents	Incorrectly Identified Car accidents	MSE
1	10	10	0	0
2	20	20	0	0
3	30	29	1	1
4	40	39	1	1
5	50	49	1	1

Table 1: Mean Square Error Measurement

An extensive experiment with the convolutional neural network (CNN) module of the car accident detection methodology yields a measured mean squared error (MSE), which is then used to compute the MSE. By calculating the square root of the mean squared error (MSE) average, we may calculate the root mean square error (RMSE) with a magnitude of 1.264. The error rate is a measure of how well the CNN model was implemented.

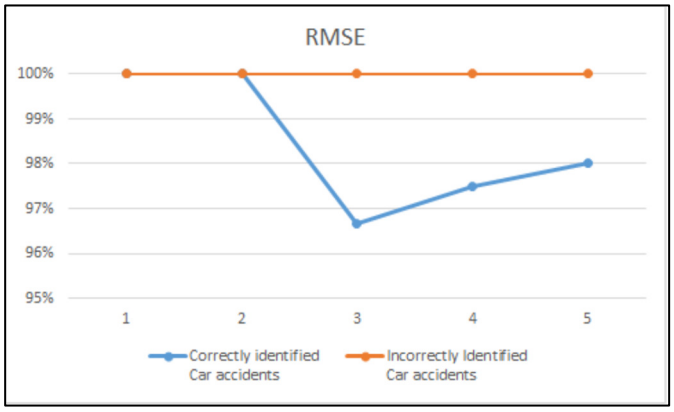

Figure 3: No of proper expected of scores V/s No of Obtained Scores

v CONCLUSION AND FUTURE SCOPE

This research details a process for automated vehicle accident investigation that makes use of deep learning techniques and produces far better results. This method takes advantage of the video evidence of every vehicle accident by using Mobile Net Neural Networks. A dataset consisting of several films of vehicle accidents is used to train the Mobile Net Neural Networks model. The dataset is first preprocessed, then the images inside it are normalized, and finally the preprocessed images are sent into the Mobile Net Neural Networks for training. After the model has been trained, it is used to test on input video that has been preprocessed and normalized but has never been tested with Mobile Net Neural Networks recognition before. A good decision-making procedure is utilized to achieve the categorization of results related to vehicle accident identification. The study article's results and discussion sections give a synopsis of the outcomes that were achieved by applying the suggested method. With such a small margin of error, the results prove that the proposed methodology is effective.

REFERENCES

[1] Dr. C. K. Gomathy, K Rohan, Bandi Mani Kiran Reddy, Dr. V Geetha "ACCIDENT DETECTION AND ALERT SYSTEM" See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/360620242

[2]V. Veera Anusuy, B. Jasmith, P. Sugasini, M. Jeeva Dharshini,"An IoT Based Accident Detection and Rescue System" ISSN: 2584-2854 https://goldncloudpublications.com\

[3] Duaa Hadi Nassar, Jamal Mustafa Al-Tuwaijari "Vehicle Accident Detection and Notification System" Iraqi Journal of Science, 2024, Vol. 65, No. 7, pp: 4061-4075 DOI:10.24996/ijs.2024.65.7.40

[4] Dr.M.Sarojini Devi ,Mrs.P.Sujidha,Mr.A.Muthuvel, Mr.H. Peer Oli "IOT Based Accident Detection System for Smart Vehicles and Prevention System for E-Vehicles" See discussions, tats, and author profiles for this publication at: https://www.researchgate.net/publication/380909879

[5] Heesoo Kim , Hyorim Han ,Yongsik You ,Min-Je Cho , Junho Hong ,and Tai-Jin Song "A Comprehensive Traffic Accident Investigation System for Identifying Causes of the Accident Involving Events with Autonomous Vehicle" Volume 2024, Article ID 9966310, 25 pages https://doi.org/10.1155/2024/9966310

[6] Andy Berres ,Pablo Moriano ,Haowen Xu , Sarah Tennille ,Lee Smith ,Jonathan Storey, Jibonananda Sanyal"A traffic accident dataset for Chattanooga, Tennessee" See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/382090017

[7] T V Pavan Kumar, Tiwari Vaishnavi, V.Lakshmi, Gaurav Gupta "Advancements in IoT Technology: A Comprehensive Approach to Accident Detection and Emergency Response" https://doi.org/10.1051/matecconf/202439201087

[8] Shehzad Aslam,Shahid Islam , Natasha Nigar , Sunday Adeola Ajagbe ,and Matthew O. Adigun"An IoT-Based Automatic Vehicle Accident Detection and Visual Situation Reporting System" Volume 2024, Article ID 4719669, 14 pages https://doi.org/10.1155/2024/4719669

[9] Sadda Bharath Reddy, Pulakandla Vivek Reddy, KaveliIndra Reddy, Kalpure Devraj, Jajimogga Sravanthi, Khristina Maksudovna Vafaeva "Android accident detection and alert system" https://doi.org/10.1051/matecconf/202439201080

[10] Tanusree Chatterjee, Priya Roy, Kamalesh Karmakar , Shiladitya Munshi ,Sanjib Roy "Deep Learning Model for Detection and Severity Analysis of Car Accidents"ISSN 0867- 6356 DOI: 10.2478/fcds-2024-0012 e-ISSN 2300-3405

[11] Nur Nazifah Adlina Mohd Nazeem, Siti Lailatul Mohd Hassan, Ili Shairah Abdul Halim, Wan Fadzlida Hanim Abdullah, Nasri Sulaiman "Microcontroller-based camera with the sound source localization for automated accident detection" Vol. 13, No. 3, September 2024, pp. 639~646 ISSN: 2252-8814, DOI: 10.11591/ijaas. v13.i3.pp639-646

[12] Chun-Hao Wang,Yue-Tian-Si Ji , Li Ruan ,Joshua Luhwago,Yin-Xuan Saw,Sokhey Kim, Tao Ruan,Li-Min Xiao ,and Rui-Jue Zhou "Multisource Accident Datasets-Driven Deep Learning-Based Traffic Accident Portrait for Accident Reasoning" Volume 2024, Article ID 8831914, 18 pages https://doi.org/10.1155/2024/8831914

[13] Deeksha K, Kavya S, Nikita J, Evangeline R. C " Road Accident Severity Detection In Smart Cities" ISSN : 2456-3307 www.ijsrcseit.com doi : https://doi.org/10.32628/CSEIT241024

[14] Victor A. Adewopo,Nelly Elsayed, Zag ElSayed, Murat Ozer, Ahmed Abdelgawad and Magdy Bayoumi "A review on action recognition for accident detection in smart city transportation systems" Adewopo et al.Journal of Electrical Systems and Inf Technol (2023) 10:57 https://doi.org/10.1186/s43067-023-00124-y

[15] Shubham Gade, Bhakti M Kholpe, Uday B Paikrao and ,Gauri Jaywant Kumbhar " Enriching redistribution of power in EV ChargingStations through Deep learning". IJSRMST | Vol. 4| Issue 1 | January2025

*****

