NORTH SOUTH UNIVERSITY

Department of Electrical and Computer Engineering

You Only Live Once: A deep learning-based approach for car crash prediction

Fahim Faisal Deepto ID- 1811899042

Fahad Bin Gias ID- 1821426042

Shakiful Alam Oyon ID- 1821512042

Faculty Advisor:

Dr. Shahnewaz Siqqique

Assistant Professor

CSE 499B, SUMMER 2022
SENIOR DESIGN PROJECT

Declaration

It is hereby acknowledged that:

- No illegitimate procedure has been practiced during the preparation of this document.
- This document does not contain any previously published material without proper citation.
- This document represents our own accomplishment while being Undergraduate Students in the North South University.

Sincerely,

Fahim Faisal Deepto
ID: 1811899042

Fahad Bin Gias
ID: 1821426042

Shakiful Alam Oyon
ID: 1821512042

Approval

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation.

Dr. Shahnewaz Siqqique
Assistant Professor
Department of Electrical and Computer Engineering
North South University
Dhaka, Bangladesh

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation.

Dr. Rajesh Palit
Professor & Chair
Department of Electrical and Computer Engineering
North South University
Dhaka, Bangladesh

Acknowledgement

We would like to begin with our gratitude towards North South University's Department of Electrical Engineering and Computer Science for providing us with the platform to showcase our design capabilities, troubleshooting ability and implementation of theoretical knowledge fed to us through the core courses designed in the program and ultimately leading to the completion of senior design project.

Our most sincere gratefulness is to our project supervisor Dr. Shahnewaz Siqqique, Assistant Professor of department of Electrical and Computer Engineering. His relentless support and motivation throughout the project term for which we shall remain indebted forever. The completion of this project would have been implausible with his support and supervision. Last, but not the least, we would like to thank our family members, friends, peers and all other personal, to whom we might have caused any inconvenience to, during the project term, for their understanding and support.

You Only Live Once:

A deep learning-based approach for car crash prediction.

Abstract:

As the number of car accidents climbs, so makes the demand for high-performance car crash detection systems. We present an automobile accident detection system that uses video data from dashboard cameras to improve car crash detection performance. Because different types of data derived from the same information source (e.g., dashboard cameras) can be considered as distinct viewpoints of the same source, the proposed vehicle collision detection system uses an ensemble deep learning model based on multi-modal data (i.e., both videos). These different views complement one another and increase detection performance since one view may have information that the other view does not. According to the experimental data, the suggested car crash detection system will be able to outperform the existing systems. We used 80 percent of the repository was used to train and the remainder to test our model. Through multiple testing and tuning of the algorithm the model was able to detect accidents with fast response time and more than 90% accuracy. The performance of the prediction algorithm fell short due to several limitations in our dataset. Upon successful implementation this system could have a significant socio-economic and environmental impact and pave way for innovative thinking.

Table of Contents

1. Introduction

1.1 Introduction

It is impossible that individuals around the world do not hear about traffic accidents every day, especially those living in Bangladesh. According to World Health Organization's latest report 93% of the world's fatalities on the roads occur in low- and middle-income countries [1]. On a daily basis, road accidents kill thousands of people and destroy invaluable properties without discrimination. It is one of the world's most dangerous incidents, causing death and property destruction. As the general assembly of the United Nations has set a target to halve the number of deaths and injuries from road traffic accidents by 2030, we hope to contribute to achieving this objective with our proposed solution. Amidst all the existing solutions and strategies out there to save life and property from road accidents, we felt the need to create a model to learn from our mistakes.

1.2 Problem Statement

There exist numerous solution and technology developed to address the problem in discussion. Identifying the key causes of road accidents will allow for the development of a solution that reduces the detrimental effect of severity on human and property loss. Road Severity does not develop by coincidence; it has predictable tendencies and may be avoided. Thus, accidents are "observable, measurable, and preventable events" [1]. There are a few approaches that use neural network to solve the problem. To name a few; there is a technology to detect pothole and speed breakers using Convolutional Neural Networks (CNN) [2], there is another pothole detection technology using machine learning classification algorithms [3]. Furthermore, there is even a prevention system that focuses on the driver and based on his/her

behavior gives necessary safety instructions [4]. These are merely a few of the numerous technologies already being developed. These strategies aid in gaining insight and identifying the root cause of car accidents and other issues affecting road safety. However, no approach exists that incorporates all of these technologies. Moreover, there are only a few of technologies that are capable of predicting accidents, and none of them are effective enough to be deployed. Each of the existing safety measures is adequate on its own. However, if we could merge these existing technologies into a holistic approach, the result would be superior and much more effective. So, our objective is to develop a neural network-based model capable of detecting and predicting accidents caused by anomalous car movement or speed, close proximity between vehicles, poor road conditions, and adverse weather conditions in order to prevent them.

1.3 Proposed solution

It is famously said by Winston Churchill, "Those who do not learn from history are doomed to repeat it." Past mistakes may not give us a program for the future, but identifying the common traits or errors on the road will help us face such situations better in the future. Therefore, in our suggested system, the model will acquire knowledge from numerous video clips of accidents that have occurred in various scenarios. Out of numerous sources, we want to use videos recorded in the dashboard camera of cars. In order for our model to forecast a potential accident, it must first recognize an accident. Our video dataset will be utilized for training a detection algorithm. Then, once the model has learned the accident patterns, it will be able to make predictions based on our methodology. We aim to employ a real-time detection algorithm so that we may generate an output as promptly as possible.

2. Literature review

The first model we studied, intended to develop a car crash detection system using ensemble deep learning and multimodal data from dashboard cameras [5]. It collected data from Youtube. They developed a system to detect accidents using both audio and video data. To achieve this, they developed a separate GRU-and-CNN based for audio and video and merged them together to detect accidents. Hence a state-of-the-art classification model was developed. This paper helped us train our model to learn what a road crash is. The detection modeling methods used in the paper gave us a ground to make a more promising prediction system to achieve our goal.

In another work, there was a proposed modeling of a vehicle accident prediction system based on a correlation of heterogeneous sources [6]. They gathered all of the accident-related data and used unsupervised learning with K-means clustering to group the highly responsible factors. The elbow method was used to find the high-risk clusters. As a result, the top 6 factors of accidents were identified beyond vehicle and driver condition. The model also analyzed the correlation between these factors in predicting accidents. This paper helped us tremendously to identify the factors that can lead to a road crash. Also, the analysis of the correlation between the factors enabled us to develop a more accurate crash prediction system. As the top factors have been identified, we can now apply hyper-parameter tuning to further find the highly effective features for our prediction model.

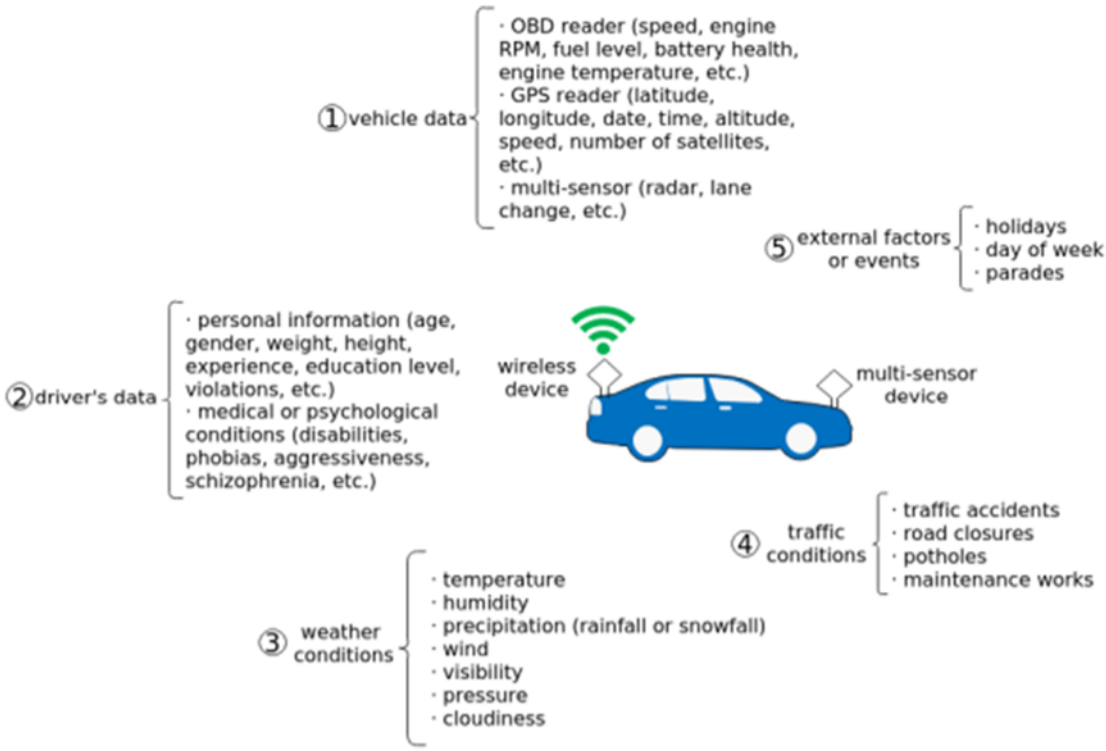


Fig: Heterogeneous data sources

In the context of Bangladesh, a highly classified model for pothole and speed breaker detection using smartphone cameras and convolutional neural networks was developed [2]. The paper used a custom-built dataset focusing on developing countries like Bangladesh. It explored and applied Faster R-CNN to develop a highly accurate classification model. It trained the model with both images and video. This high-tech model can find potholes and speed bumps with a very high level of accuracy not only from images and recorded videos, but also from live video data. In underdeveloped countries such as Bangladesh, poor road conditions are a major role in accidents. This advanced categorization model enabled us to create a more accurate model for recognizing road anomalies. In addition, the database is constructed with Bangladesh's traffic

conditions in mind, which differ significantly from those in affluent nations. As a result, we learned a great deal about the road conditions.

Another paper recognized that identifying unusual occurrences, such as traffic offenses and accidents, in typical driving scenarios is crucial for the success of autonomous driving and sophisticated driver support systems [7]. However, the vast majority of work on video anomaly identification is plagued by two significant shortcomings. First, they presume cameras are stationary, and recordings have static backgrounds, which is appropriate for surveillance applications but not for cameras placed on moving vehicles. Second, they frame the issue as a one-class classification, relying on painstakingly hand-labeled training datasets that restrict identification to specifically defined anomalous categories. This work provides an unsupervised method for accident identification in first-person (dashboard-mounted camera) footage. Their primary innovation is discovering abnormalities by forecasting the future locations of traffic participants and monitoring the prediction accuracy and consistency parameters using three distinct approaches. They analyzed their method using a novel dataset of different traffic incidents, AnAn Accident Detection (A3D), in addition to a publically accessible dataset. Their method outperforms the state-of-the-art, as shown by experimental data.

A computer vision approach for lane detection and tracking was proposed in research where the system assists the driver in determining the proper lane, and an alarm is shown on the screen if the vehicle deviates from the correct lane [8]. Besides displaying the vehicle's location relative to the center of the lane, the screen shows the radius of curvature at each place compatible with the road. The suggested device attaches a camera to the car's dashboard, allowing the camera to capture video of the road ahead easily. This technique is helpful for drivers to prevent collisions and may also be used by autonomous cars to follow specific lanes.

This method's multiple benefits highlight that this is often a promising strategy. Considering that most automobiles are already equipped with a front-mounted camera, the costs associated with this method are pretty minimal. This system considers two methods; the first is based on Canny edge detection and the Hough transform, while the second is based on the Sobel operator and perspective transform. The second method proved more accurate and precise in recognizing the lanes and bends within these lanes. So, the final system was made with the help of the Sobel operator and the perspective transform.

A system for pedestrians and vehicles recognition used a laser range finder and a webcam, which provides drivers with information on driving safety on a display device [9]. The laser rangefinder gathers information on the distance and azimuth angle of the surrounding environment, which detects the breakpoints of objects in the vicinity of the vehicle. In addition, Haar detection is employed to extract the relevant characteristics of people and automobiles for picture identification systems. The detection of people and cars on the road was more precise when an image recognition system was coupled with laser distance information. According to the testing findings, the suggested system correctly recognizes people and automobiles.

In an attempt to detect unsafe driving patterns, it was determined that over-steering to turn over, not steering enough to stay in lane, over speed, high located center of gravity, weather conditions, road conditions, and the road's curves are the most contributing factors to a heavy-goods-vehicle (HGV) overturning [10]. Hence, this research presents machine learning algorithms to address these issues and minimize HGV rollovers. The suggested system consists of a vehicle-mounted device and an operational surveillance camera located on the ground. The car-equipped system can assess the safe speed at which the vehicle should travel based on the kind of vehicle and the road's curvature, as well as identify road cracks and alert the driver

through the dashboard display. The ground-based driver assistance system can figure out how fast an HGV can go safely and what other traffic conditions could cause an HGV to roll over.

A project attempted to minimize the incidence of accidents by creating a device with the help of embedded technologies and the Internet of Things that can be fitted into any vehicle without altering its generic design [11]. People's difficulties have been considered, such as implementing the essential tools, features, and functionalities to lower the alarming accident rate. The system included automated identification of the distance between cars in front when they appear in close proximity, the configuration of an automatic brake triggering assembly, and an alarm system. Additional capabilities include live tracking, WIFI Fidelity in automobiles, and dashboard camera surveillance. This component has increased the vehicle's security and deters theft. The main benefits of this system are that it helps the owner keep an eye on the car faster than other systems, is less complicated than other systems, and is the cheapest and safest option for an anti-theft system.

We also studied a model that uses dash-board camera videos like ours. It was found in the study that video-based traffic monitoring systems are widely used to detect accidents in cities. However, it is restricted to the surveillance region. To circumvent this constraint, they present a method for identifying accidents from dashboard camera footage via computer vision-based algorithms, analyzing the variance in visual information during accidents in order to create an effective accident detection model with a lower computing cost [12]. Consequently, accidents are recognized based on the variation in visual information that occurs during an accident. It is assessed using the order to develop a new Dashboard Video Accident Detection (DVAD) dataset. The experimental investigation shows that the suggested method for spotting accidents works. This method could be expanded to let the right people know when an accident happens.

Lastly, as we are inching towards the era of autonomous vehicles, we wanted to see the scope of our model in that environment. A study presented a novel support system for autonomous vehicles that uses a dashboard camera to detect collisions [13]. The solution uses the Mask R-CNN framework for vehicle recognition and a centroid tracking algorithm for vehicle tracking. In addition, the framework evaluates numerous metrics, such as velocity, acceleration, and trajectory, to determine if a collision has occurred between any of the tracked vehicles. On a certain set of dash-cam footage, the framework can find accidents with a high rate while keeping the number of false alarms to a low level.

3. System Design

In order to limit the occurrence of accidents, it is intended to take a holistic strategy. To accomplish this, a robust model of the system will be constructed. It is believed that the method will enable the prediction of accidents prior to their occurrence. All potential factors that may lead or contribute to a road accident will be considered to ensure a comprehensive solution. The system will be constructed utilizing video data collected from multiple sources, and an ensemble method called YOLO (You Only Look Once) will be used to train the model. The YOLO algorithm will be able to provide us with a prediction after a single run.

3.1 Methodology

The approach shown in figure below is followed to design our system. As our system is a neural network based model, the design will be in two steps- (i) collecting and preprocessing video footages of accidents and (ii) training a model using our dataset.

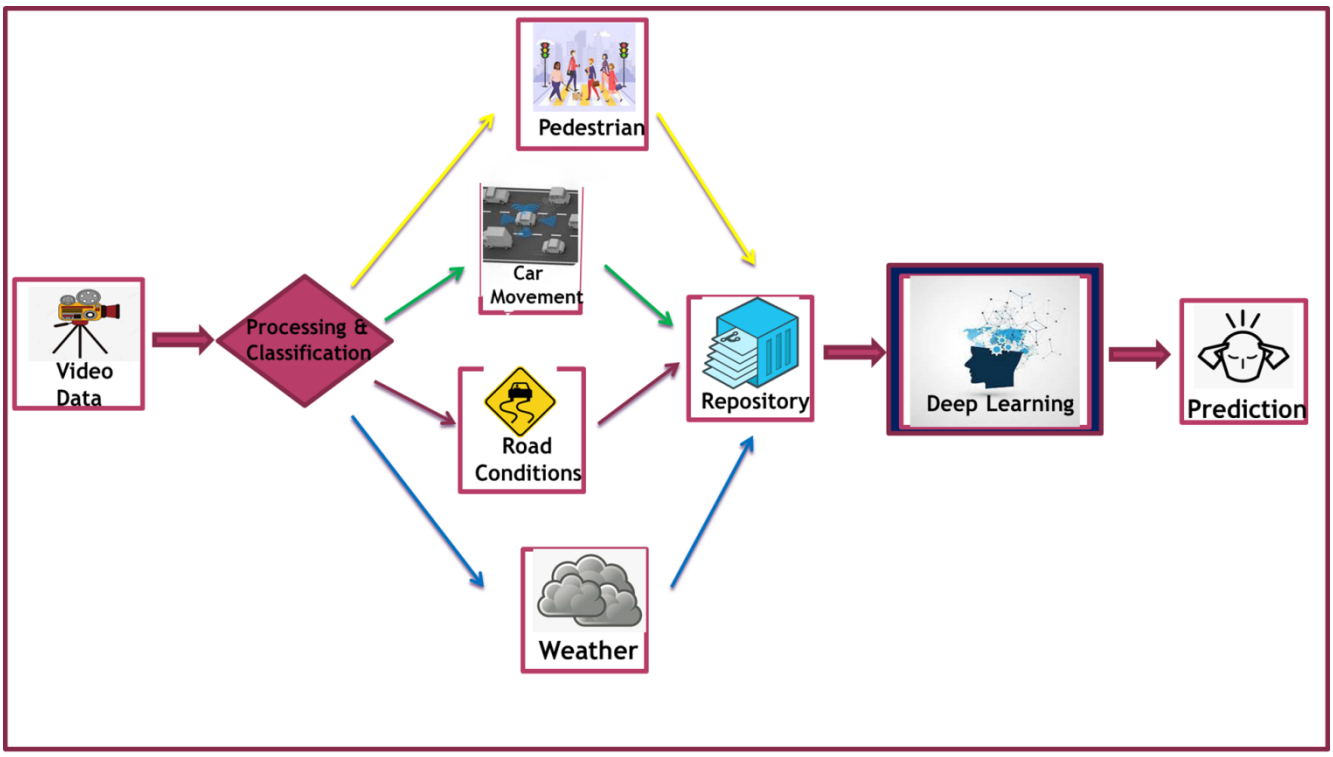


Fig: System Diagram

3.2 Obtaining data and preprocessing

At first, we gathered dashboard camera footage of accidents from a variety of online sources. This system will be implemented in Bangladesh. Since the road conditions and traffic scenarios in Bangladesh are vastly different from those in other developed nations, we constructed a dataset that encompassed the most similar scenarios possible. The majority of the dataset consisted comprised accident footage from neighboring India, as Bangladesh's traffic conditions are sufficiently similar to those of India. Taiwanese footage made up a significant portion of the data we collected. During preprocessing, we discovered that the Taiwanese dataset included a significant number of motorcycle accidents. Therefore, we chose to include them in order to ensure that the model is aware of all potential situations. The videos were disassembled frame-by-frame. Consequently, each frame was viewed as an image. Before the data gathered from the movies could be used for training, it was subjected to rigorous processing. Additionally, because the data may contain a large number of image classes, we only categorized the images required for our job. The image classes that of our concern were-

- Vehicles
- Pedestrians
- Abnormalities on the road

After being done with necessary preprocessing and classification, as it is shown in fig-1, we labeled and divided our data into four main scenarios based on the dataset we got-

- Pedestrian movement
- Car movement
- Road conditions

- Weather conditions

Although there are a number of other circumstances in which an accident may occur, we were only able to collect sufficient footage to train our system model on these three cases. We were unable to train our algorithm to recognize other types of accidents due to a lack of credible data. At this point we have a sizable amount of data ready for training and testing a neural network model.

3.3 Training the Model and Creating Prediction System

After preprocessing, we had our dataset to train the model. But Deep Learning algorithms understands only number. So we applied Bounding Box attributes to the images using YOLO. This classified all the objects inside our video dataset and our dataset will be finally ready. Then using YOLO algorithm the dataset was trained to create a model. A small portion of the dataset was kept for testing the model later. Now this would still only deliver object detection and position of the objects in the images. From the data we obtained from various accident scenarios, we will then apply constraints on our model. By doing so our model will learn upon which conditions it has to predict that an accident might happen. Thus, based on the collected data, our model will be able to make an accident predictable and hence avoidable. Since the connections in among the layers of our neural network were very dense, we used DenseNet to preserve the feed-forward nature. This helped to prevent information from vanishing between layers. Later, the entire model was tested by the test set of our dataset. We continuously improved and tuned our algorithm until we got an acceptable result.

3.3.1. Pedestrian movement

In YOLO v3, the bounding box co-ordinates are determined by splitting the image into a S x S grid, where one grid cell is responsible for anticipating each object present in the image, which is the grid cell in which the object falls into, which is the object's center. The bounding box predictions include four components: (centerX, centerY, width, and height), with centerX and centerY indicating the box's center position. The formula x = centerX - (width / 2) y = (centerY - (height / 2) is then used to get the top-left and bottom-right coordinates of the enclosing box using the center coordinates [14]. In the figure, we can see the use of YOLO for the pedestrian movement.

3.3.2. Car movement and proximity

It's difficult to detect approaching automobiles exactly near a moving camera in continuous frames of video. During this study, two major drawbacks of the background subtraction technique in vehicle recognition with a moving camera were discovered. The first is that because the background is dynamic, it is difficult to distinguish moving objects in each frame while the camera is moving quickly. It is constantly changing in every picture. The second problem is that when the camera is moving, this algorithm picks up on things like waving trees and signs in each frame, even though all that's needed are heavy, fast-moving cars coming up to the moving camera. As a result, this system is ineffective in detecting the car in each frame. The YOLO Object Detection algorithm is employed in this proposed strategy to detect the coming vehicle in each frame. Every frame used a trained data set for automobile detection. The outcome of the vehicle detection technique is shown in the figure, where the bounding box is created on

the vehicle and its center signifies the vehicle's centroid [15]. Besides that, in this model we will do:

- Detect vehicle speed which is approaching towards the moving-camera
- The video is converted into frames
- The position of the centroid of the detected vehicle is observed over subsequent frames
- Speed is calculated using a particular distance travelled by the vehicle.

3.3.3. Road Condition

We employ pothole-filled roadway image data. Annotation and labeling is the process of identifying potholes in photos. The solution is to draw a bounding box around the pothole and identify it. Label tools aid this procedure in producing YOLO framework input formats. Every piece of research data was annotated and labeled. Id class, bounding box center (x and y), and bounding box width and height are all stored in the annotation and labeling process (w and h). The bounding box information is in decimal format on a 0–1 scale, and the id class is an integer number starting at 0. Each.jpg photograph will be accompanied by a.txt file including pothole information as we saw in the fig.9. All models will be tested using six different numbers of test data during the detection process: 224, 150, 96, 64, 56, and 26. The goal of this test is to compare the detection result to the ground truth to determine performance value. As a parameter evaluation, we use IoU (1), precision (2), recall (3), and mAP [16].

3.3.4. Weather Condition

Thermal cameras that use infrared (IR) technology record the heat created or reflected by the objects being monitored and convert the energy into temperature values to create an image.

Because thermal radiation sensors in the MWIR and LWIR ranges pick up the heat given off by the objects being viewed, cameras in these ranges (Fig. thermal image) don't need an extra light or heat source [17]. Another important aspect that influences recording quality and consequently, detection performance is the weather. The distance at which it is possible to accomplish a successful object or human detection decreases as weather conditions deteriorate. For example, in long-distance shooting, some features that may be crucial for object recognition, such as a human leg, are minuscule and are represented with only a few pixels in the image. When the weather is severe, the image quality can decrease to the point that these parts are barely visible or not viewable at all. This loss of data can have a significant impact on the classifier's performance. Both mid-range and long-range thermal camera sensors can be utilized in protected environments such as border controls or airports for 24/7 all-weather surveillance. Because different bands of sensors have varied strengths and limitations, there is no one-size-fits-all solution for all surveillance applications. Different environmental, climate, temperature, and meteorological conditions have different effects on sensor performance. Both bands are unaffected by solar radiation but are harmed by fog and rain, with the LWIR band outperforming the MWIR in foggy situations. In most climates, MWIR sensors are less impacted by humidity than LWIR sensors for most target ranges and have stronger atmospheric transmission than LWIR. Warmer temperatures benefit from MWIR sensors, whereas colder climates benefit from LWIR sensors.

4. Technology Used

4.1 Neural Network

The Neural Network is a computational learning system that interprets and translates a data input of one type into a desired output, which is typically in another form. This is accomplished through the utilization of a network of functions. The idea of an artificial neural network was conceived after being influenced by the way neurons in the human brain collaborate with one another through synapses to comprehend information received from the human senses [18]. The learning algorithm for neural networks learns by analyzing a large number of labeled instances that are provided while the network is being trained. The algorithm figures out which features of the input are necessary to build the appropriate output and then learns those features. When a sufficient number of instances have been processed by the neural network, it will then be able to start processing new inputs that it has not previously seen and successfully produce accurate results [19]. Just like the human brain, the program is able to learn from experience and become more knowledgeable as it is exposed to more examples and a wider variety of inputs, the usual accuracy of the program's outputs tends to improve. A neural network is a multi-tiered system with multiple nodes on each layer. Each layer is highly interconnected. Using a neural network has the advantage of being adaptive. This means that as more information is continuously provided, the linking layers evolve from their initial condition. A rudimentary neural network is shown in figure below. Neurons are the nodes that comprise the layers. Between input and output, the neural network is composed of layers of function. The diagram illustrates the interlayer interactions of neurons.

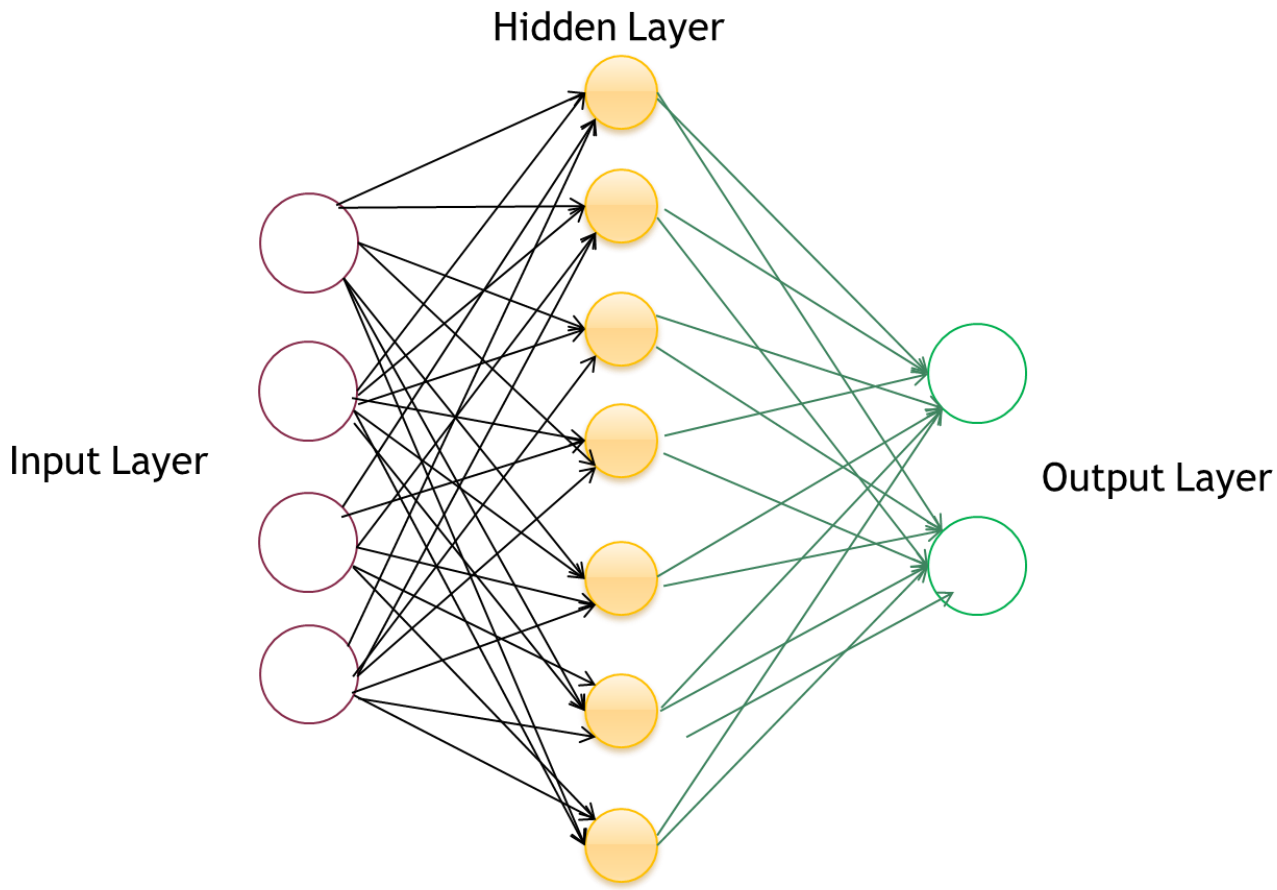


Fig: A Rudimentary Neuron

4.2 YOLO

You Only Look Once; commonly known as YOLO is a real-time object detection system based on the darknet architecture that is state-of-the-art. A typical video contains 24 frames per second (fps), and real-time detection necessitates an algorithm that is faster than this. The algorithm ensures faster outcome with compromising a little bit accuracy. YOLO improves detection performance by training on entire photos. This unified model has various advantages over standard object detection approaches. First

and foremost, YOLO is lightning fast. On a Pascal Titan X, YOLOv3 processes images at 30 frames per second and has a mAP (mean Average Precision) of 57.9% on COCO test-dev(2); a faster version operates at 150 frames per second (with less mAP). YOLOv3 is 100 times quicker than fast RCNN and 1000 times faster than RCNN while maintaining the same level of accuracy, making it ideal for real-time object recognition [20]. YOLO comes from the fact that the entire forecast is made in a single image evaluation. We don't require a complicated pipeline because we define detection as a regression problem. To forecast detections, we simply execute our neural network on a new image at test time. We don't require a complicated pipeline because we define detection as a regression problem. To forecast detections, we simply execute our neural network on a new image at test time [21]. Furthermore, while making predictions, YOLO considers the image as a whole. YOLO sees the full image during training and testing, unlike sliding window and region proposal-based approaches, therefore it implicitly encodes contextual information about classes as well as their appearance. So not only we get to know if the object is there but also the position of the object inside the image. The working procedure of YOLO is-

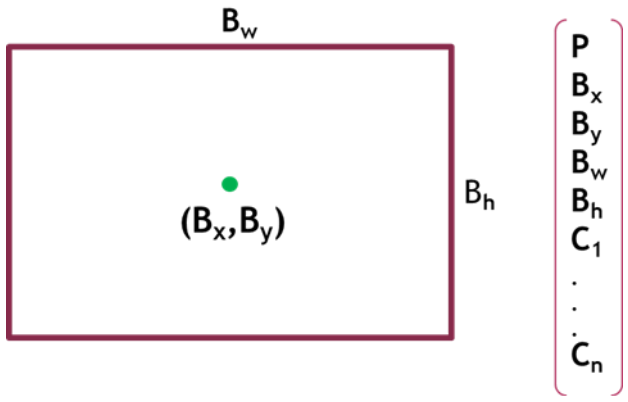


Fig: Bounding box

YOLO divides an image into grids and each grid is responsible to detect object inside them. As it detects an object in an image it creates a bounding box for the object. Each object class in an image has a bounding box like in figure below.

Each bounding box has a vector like in above figure which represents several attributes about the object detected inside the bounding box.

- **P**= Probability of a class being in the box (value can be 0 to 1).
- **(B_x, B_y)**= Center of the bounding box
- **B_w**= Width of the bounding box
- **B_h** = Height of the box
- **C_1 ….. C_n** = Classes of image(1 if the image class is present, 0 if not)

If there is only a single object in an image like in below picture, then the bounding box will detect the object and in the matrix probability variable will be P=1 and class detection variable C=1. From these values we can say whether or not the object is in the

present. But with YOLO we can get more information as we know the center of the object's bounding box as well as the height and width. These values will give us the position of the object inside the image.

Fig: Single object in an image

Now, when there are multiple objects in an image, it means there will be multiple classes (C_1,…C_n). YOLO algorithm handles this step by step. Let say there are two objects of class 1(C_1) (C_1,…C_n). YOLO algorithm handles this step by step. Let say there are two objects of class 1(C_1) = Car and class 2(C_2) = Human. First, it divides the image into grids like in figure below. Each grid here will be responsible for detecting any object inside them.

If we take a grid that has none of the object class defined. Then the probability variable in vector will be P=0 and since there is no object no need of class type. On the other hand, taking a grid with car in it will return the vector with values; P=1 and $C_1 = 1$, as there is a car class object inside it. Simultaneously, from the remaining values of the vector, we will get the position of it in the image. But, what if YOLO predicts two bounding boxes like in figure below?

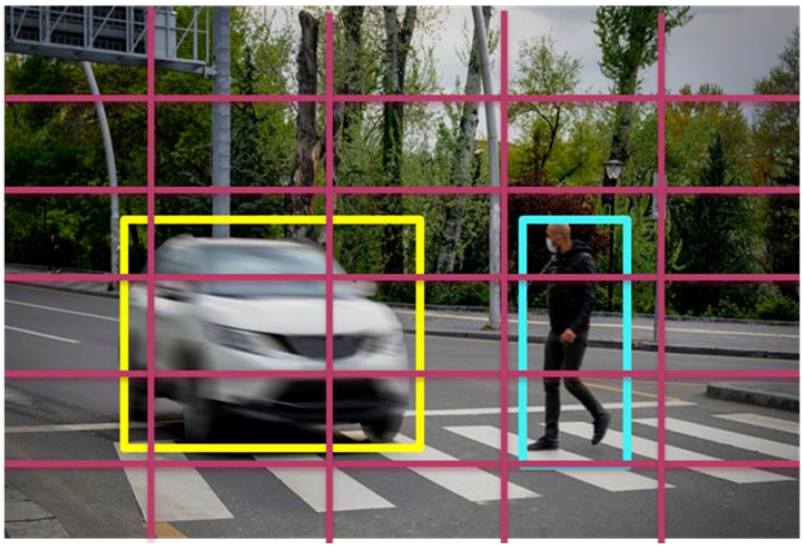


Fig: Multiple objects in an image

Fig: Grid without any object class

Fig: Grid without object class

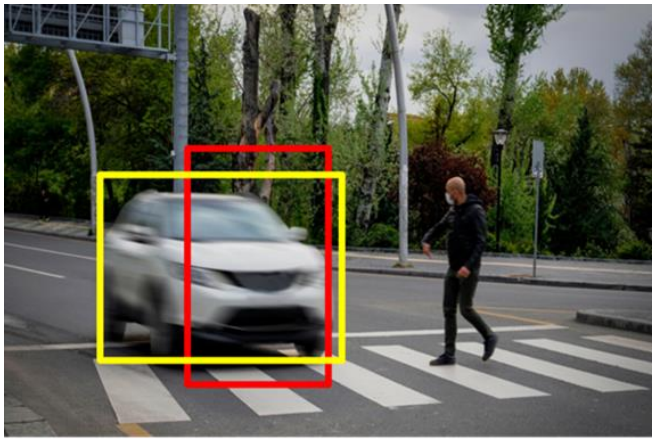


Fig: Multiple bounding boxes

YOLO tackles the problem with the concept of Intersection over Union. Which means the algorithm will calculate the intersection area of these two bounding boxes and divide it by union area of these two boxes to get the prediction as well as the position of the object in the image. YOLO performs all these steps in one go which means it takes less time to train a model using YOLO.

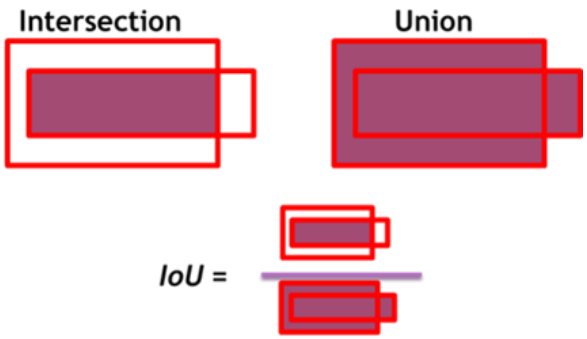


Fig: Intersection over Union

4.3 DenseNet

Each layer of the neural network in DenseNet receives additional inputs from the layers that precede it. The inputs are then concatenated and mapped onto the layer's attributes. Therefore, this common knowledge is conveyed to all succeeding layers. As each layer receives a feature map from all preceding layers, fewer channels are required, resulting in a more compact network.

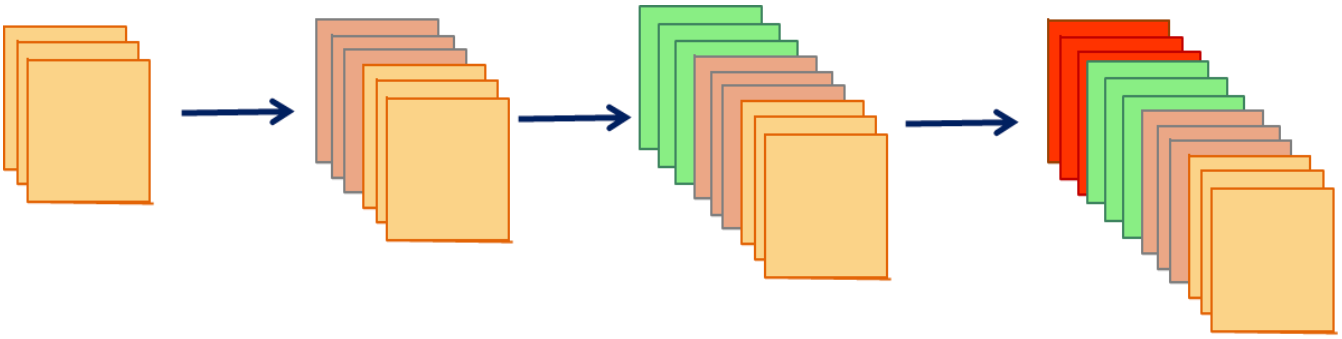


Fig: DenseNet Architecture

The result is a more computational and memory efficient system [22]. For each layer's composition, Batch Norm, ReLu, and finally Convolution are applied to the output feature maps. Inside a dense block, feature map sizes are same, making concatenation easy [22].

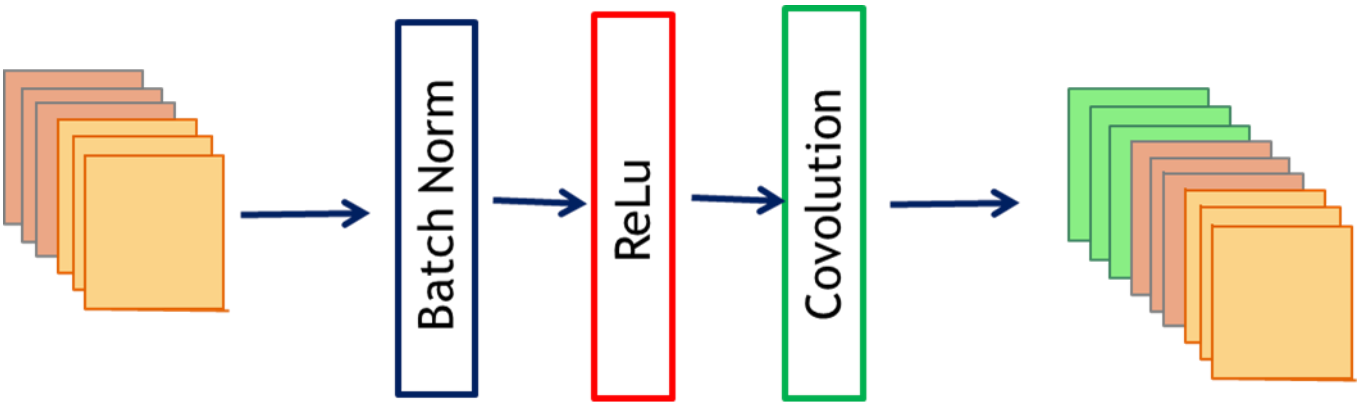


Fig: Between DenseNet layers

4.4 Advantages of our Approach

We have decided on our proposed methodology as this approach and the technology we are using gives us certain advantages in achieving our goal. Some of them are-

- Neural Networks are adaptive and hence it will be able to learn continuously with changing scenarios.
- YOLO algorithm does objective detection in a single run. It is faster and optimal for real time object detection especially for processing video data. We working with a case where timing is key.
- DenseNet has very strong gradient flow [22]. Provides better computational efficiency in comparison to many. As DenseNet passes the feature maps onto the proceeding layers, it prevents the model from losing information mid-layer. Lastly DenseNet gives a smoother decision boundaries [22] and so it can perform well even if there are insufficient training data.

5. Result & Analysis

5.1 Result

We obtained separate results and findings for accident detection and prediction using our trained model. Our test set consisted of 20% of all the data we collected. We trained our dataset with the YOLO real-time technique. The obtained model has an overall accuracy of 92% on a randomly selected test set after completion of training. The detection model was 90% effective at identifying accidents and 93% effective at identifying a scenario with no accidents. Due to the fact that there are just two classes to detect and there is no class imbalance, we chose test accuracy as the matrix to examine our results. The system's other component is accident prediction. Along with its detection counterpart, the prediction algorithm has a success rate of approximately 60% on unobserved test data. As is evident, our prediction system requires additional tuning to solve the model's deficiencies. In addition, our inability to precisely measure vehicle speed is a significant contributor to the low accuracy of our predictions. As our system requires quicker and more precise outcomes, we were able to meet the time requirement. Due to the employment of a real-time algorithm and DenseNet to improve outputs, the system was able to provide a result on previously unknown data in a short amount of time.

5.2 Analysis

Our model was able to deliver in terms of speed and detection. It was able to detect accidents or non-accident scenarios within a very fast time. However the lacking is in predicting an accident. Through the problem solving approaches we took to solve this, we have identified the key reason for such average predicting results. Our Hypothesis is that majority of the wrong predictions are coming when a scenario with measuring car proximity is at hand. By running several tests and tuning the prediction algorithm, we believe that our approach in measuring the car proximity is

not efficient enough. The reason is not only our technical limitation but also the fact that majority of our dataset are dashboard footages. It is very difficult to measure distance of another vehicle using dash-board camera. Based on the vehicles size, dashboard cameras can be mounted at different angle. So much diversified scenarios for the same situation is what is holding us back. Hence, we need to further tune our algorithm so that we could overcome the fundamental flaw in our dataset can be avoided.

5.3 Technological performance analysis

In the following section, we will talk about the many models that may be used to identify objects while maintaining the necessary level of accuracy quickly. We test each model with the identical kind of video data in order to determine which one is the best at recognizing whether a car has been involved in an accident. in order for us to accurately determine the real model for the detecting process. Here, we need to keep a few things in mind, the most important one being that we are going to employ the most efficient model for auto accident detection in our future plan, which is entitled "You only live once." Within the scope of that research, we will make use of this model to determine an appropriate prediction model for the driver of a vehicle or bike. Here, initially, we compile a wide variety of video data that is comparable to the situation in our nation. First, we employ a manual filtering process to separate the CCTV footage from the rest of the movie. According to the findings of the Background study, people have achieved a satisfactory level of accuracy in the detection process by making use of CCTV footage. However, CCTV footage cannot be included in the prediction model for the foreseeable future. It is possible to utilize it as part of an emergency call project to save someone injured in an accident.

SVM classifier is what we utilize in this place for video pre-processing. After we were through with the preprocessing, we trained our data on a number of different models to see which models had a high level of accuracy when it comes to detecting vehicle accidents.

The YOLO-CA training outcomes, including the changes in accuracy, recall, IoU, and loss that occurred throughout each batch iteration procedure. During the YOLO-CA training procedure, we consider the prediction result to be a true result if it has an IoU that is more than 0.5 and the correct classification. All other predictions are considered to be false results. As may be seen in the table, the outcomes of the predictions may be broken down into the following categories:

1) TP stands for "Truth Positive."
2) FP: Shorthand for "false positive."
3) Also known as a false negative
4) TN: Shorthand for "True Negative."

The formula for calculating precision is as follows:

precision = TPTP + FP, while the formula for calculating recall is as follows: recall = TPTP + FN. During the YOLO-CA training procedure, we consider the prediction result to be a true result if it has an IoU that is more than 0.5 and the correct classification [23]. All other predictions are considered to be false results.. The formula for calculating the precision is as follows: precision = TP + FP, while the formula for calculating the recall is as follows: recall = TP + FN. As may be seen in Figure. The precision of YOLO-CA is improving progressively along with the increase in the number of iterations, and it has converged to around 90 percent. In addition, recollection ultimately reaches a point where it is more than 95 percent.

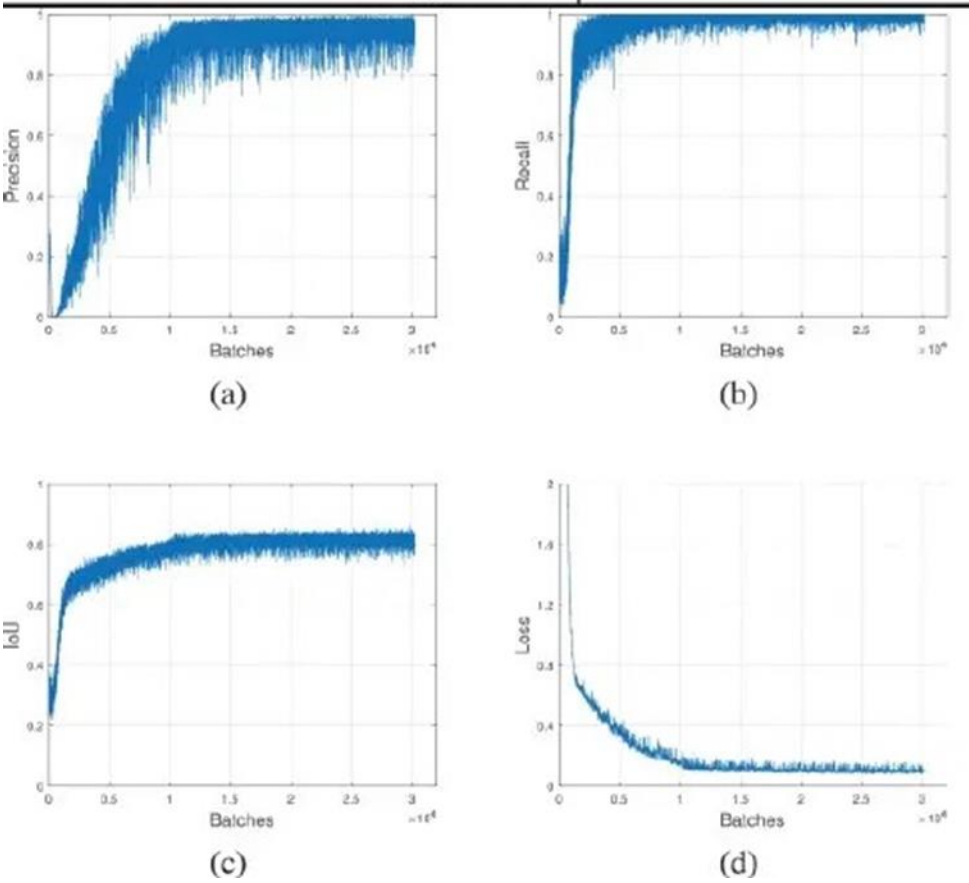
Experiments in comparative research are carried out in order to evaluate seven different detection models: (1) Models with only one stage, including SSD, our suggested YOLO-CA, the standard YOLO-v3, and YOLO-v3 with no MSFF (Multi-Scale FeatureFusion). (2) Two-stage models, including fast R-CNN, faster R-CNN, and fast R-CNN with forwarding propagation of noise. The following indices have been chosen for comparison among the seven models in order to illustrate the comparative validity of YOLO-CA as well as affirm its strength in terms of comprehensive performance in accuracy and real-time [23].

• Average Precision (AP) is defined as the average value of precision under varying recall, which may be altered by modifying the confidence threshold. The AP index measures the accuracy of detection models. Calculating the average precision is possible.

AP =∑precision(r) where r is recall.

•Average Intersect over Union (Average IoU) is used to measure the object-finding performance of a computer vision system detection models. The Average IoU is the mean of the IoUs between each predicted bounding box and the ground truth. Frames Per Second (FPS) (FPS). Inference time is defined as the average time cost of finding a frame among \stest set. FPS is the inverse of inference time, which is defined as the average number of frames per second that may be identified.

		Ground truth	
		Positive	Negative
Prediction Result	Positive	TP	FP
	Negative	FN	TN



After identifying the model that was the most successful in identifying the vehicle accident. Now, in order to use denseNet with the YOLO. We use the Dense Convolutional Network, often known as DenseNet, which is a type of neural network that connects each layer to every other layer using a feed-forward structure. Our network features L(L+1)/2 direct connections, in contrast to the L connections that are present in conventional convolutional networks with L layers, which are one between each layer and the layer that comes after it. As inputs for each layer, the feature maps of all of the layers that came before it is utilized, and the feature maps of the current layer are used as inputs for all of the layers that come after it. DenseNets have a

number of compelling benefits, including the alleviation of the vanishing-gradient problem, the strengthening of feature propagation, the encouragement of feature reuse, and a significant reduction in the number of parameters.

Our model achieves an accuracy of classification of 86.0 percent in a randomly picked test set that is constituted of 20 percent of the entire quantity of video sequences from our datasets. We will be able to simulate collisions caused by other cars, pedestrians, and the conditions of the road if we are able to retrain the model using additional data and the GPU.

However, the overall performance of the prediction algorithm that we are deploying is lacking. The primary reason for this is that we are unsuccessful in measuring car speed from the dashcam videos. In order to address this issue, our goal is to improve the efficacy of our detection system, accurately measure the vehicle speed, and then tune the parameters in order to achieve an acceptable probability of prediction.

6. System Impact

We should not lose sight of the fact that we wish to serve people with our system, even while our goal is to advance a new technology for tackling a very critical issue. Our technology, like any other, can have both beneficial and harmful effects. It is our responsibility to address both issues and be mindful of the system's impacts.

6.1 Impact on Environment

Accidents caused by vehicles can have a negative effect on the surrounding environment. Accidents frequently result in gas and fluid leaks, which release potentially hazardous substances into the environment. These compounds can poison nearby plants and be damaging to wildlife. When there is an accident, one of the most serious concerns is the possibility of oil spills from damaged automobiles. Accidents that take place in close proximity to water might do significantly more damage than usual due to the ease with which water can transmit disease. After an accident that renders a vehicle a total loss, the majority of insurance companies conclude that it is more cost-effective to replace the vehicle than attempt to repair it. Although it is possible to recycle the bulk of the auto parts, most of the time, old cars and auto parts are dumped in landfills [24]. There is a possibility that these waste compounds, which can take thousands of years to disintegrate, will make their way into our food chain. This pollution has the potential to easily harm an entire ecosystem. Accidents on the road can cause significant damage to the roadways. The repair work being done on certain roads puts an increased strain on the surrounding ecosystem. As we get closer to building a system that can predict and avoid accidents, there is reason to be hopeful that the enormous toll that accidents on the road exact on

the environment may begin to lessen. There is a possibility that our system will have a detrimental effect on the surrounding ecosystem. It is possible that there will be more cars on the road as a result of our efforts to reduce the number of times that accidents occur. As a direct consequence of this, the carbon footprint left in nature will grow.

6.2 Impact on Society

It is more common for people who come from families with lower socioeconomic status to be involved in traffic accidents. Even more significantly, it is the top cause of death among children and young people, defined as those who are between the ages of 5 and 29 years old [25]. It is impossible to put a price on the damage done to a nation when otherwise capable young adults are needlessly killed on the nation's roadways.

In 2015, there was an average of 1.8 deaths per day, and 59 persons had their lives or health negatively impacted in some way [26]. The people who are directly impacted by traffic incidents are the individuals who are injured and their families. The victims, in particular, are negatively affected by health repercussions ranging from minor to catastrophic. This is not the end of the story. It is imperative that we do not lose sight of the fact that direct participants and their families are not the only people directly impacted emotionally by the psychological fallout of the consequences of road traffic accidents [26]. In the aftermath of an accident, victims can be beheaded to some extent, which can lead to them losing their jobs or becoming unable to work again in the future. This can be a devastating consequence. Automobile accidents can result in a significant number of life-threatening injuries and the repercussions of these accidents are felt all across society. Our system can be absolutely vital to bring down the negative impact of road accidents on our society. This will ensure a healthy and prosperous social life for all.

6.3 Impact on the economy

According to the findings of a study, the entire cost of accidents in Bangladesh in the year 2020 was approximately 4,118 million USD (BDT- 35,001 crore), which is equivalent to 1.3% of the country's total Gross Domestic Product (GDP) for that year [27]. Even if we consider a developed nation such as the United States of America, the situation is not significantly improved. According to a research by the National Highway Traffic Safety Administration of the United States Department of Transportation, motor vehicle collisions cost the United States alone almost one trillion dollars every single year. In addition, the study from the Department of Transportation claims that $594 billion is spent annually on "damage from the loss of life as well as the agony and poor quality of life due to injuries." A further $277 billion goes toward the actual cost of the accidents, while additional billions go toward things like missed productivity at work, property damage, and traffic-related congestion. The actual cost of the accidents is estimated to be $277 billion [24]. Amidst all these, an innovation like ours could bring hope. A new study that was funded by Bloomberg Philanthropies and conducted by the World Bank quantifies the ways in which investments in road safety are also investments in human capital. According to the findings of the study, nations that do not invest in improving road safety stand to lose between 7% and 22% of their potential annual increase in per capita GDP over a period of twenty-four years [28]. For this reason, authorities need to give priority to expenditures that have already been proved to improve road safety. The price of delay is almost 1.25 million fatalities each year throughout the globe, decreased productivity, and lost economic possibilities. Therefore, the choice that we made to deal with the issue rather than ignore it can be beneficial to the overall interests of the country. Although it's possible that our system on its own won't be sufficient to finish the work, we can guarantee that it will be a step in the right direction and that it will motivate many people to think of their very own creative solution.

7. Future Plan

We plan to train our prediction model with more data to get the same high accuracy as our detection model, which we could not do due to our lack of resources. Upon achieving our desired accuracy, we aim to deploy an IoT device that will assist drivers in preventing road crashes. The device will be integrated with the dashboard camera along with a display, buzzer, and speaker. Live videos collected from the dashcam will be analyzed in the device in real-time and show the probability of an accident on display. Whenever the probability crosses a certain threshold, the buzzer will go on while the speaker will announce the reason for its prediction. This will help the driver take the necessary actions and prevent accidents. Initially, we intend to install the device on cars; later, we will scale up the device for other popular vehicles in Bangladesh, e.g., CNG, Motorcycle, Bus, Truck, etc. In addition, if an accident still occurs, our device will send a message to the nearby hospital, ambulance driver, and police station, as well as to the injured driver's emergency and a nearby contact. This way, our device will not only help prevent road accidents but also help reduce casualties due to road accidents. Our ultimate goal is to make the device "You Only Live Once" as affordable as possible and install it countrywide on all vehicles to make our roads safer and protect people from the dreadful consequences of road accidents.

8. Conclusion

Our goal was to develop a cutting-edge technology that could detect the likelihood of an accident occurring and so prevent it from happening. Utilizing available resources and fine-tuning state of the art real-time detection algorithms, we have developed a detection system capable of producing instantaneous output. Due to the fact that our system is based on the concepts of neural networks, we can assert that our model will continue to improve as additional data is added. However, findings from the prediction algorithm fall short of expectations due to data and technological limitations. Since we have identified the primary cause for such a bad prediction outcome, it is only a matter of time before it is resolved by making the necessary adjustments. After the prediction performance meets expectations, we want to integrate it into a smart device for deployment. Our prediction model may not be sufficient to resolve such a mammoth issue like road accident, but we can only hope that by advancing our plan with this concept, we will be able to make modest to significant effects on the socioeconomic landscape of our nation.

Bibliography

[1] William Gissane, "Accidents—A Modern Epidemic," *Journal of the Institute of Health Education*, vol. 3, no. 1, pp. 16-18, 1965, DOI: 10.1080/03073289.1965.10799649.

[2] Zahid Hasan, Samsoon Nahar Shampa, Tasmia Rahman Shahidi, and Shahnewaz Siddique, "Pothole and Speed Breaker Detection Using Smartphone Cameras and Convolutional Neural Networks," in *2020 IEEE Region 10 Symposium (TENSYMP)*, 2020, DOI: 10.1109/TENSYMP50017.2020.9230587.

[3] A.K.M. Jobayer Al Masud, Saraban Tasnim Sharin, Khandokar Farhan Tanvir Shawon, and Zakia Zaman, "Pothole Detection Using Machine Learning Algorithms," in *2021 15th International Conference on Signal Processing and Communication Systems (ICSPCS)*, 2021, pp. 1-5, doi: 10.1109/ICSPCS53099.2021.9660216.

[4] R. Mahajan, D. Prabhu and D. Ghose, J. M. Kumar, "Cost effective road accident prevention system," in *2nd International Conference on Contemporary Computing and Informatics (IC3I)*, 2016, pp. 353-357, doi: 10.1109/IC3I.2016.7917988.

[5] Chan Woo Konga, Gyeongho Kima, Sunghoon Lim Jae Gyeong Choia, "Car crash detection using ensemble deep learning and multimodal data from dashboard cameras," *Expert Systems with Applications*, 2021.

[6] Lorena Barona, Leonardo Valdivieso, Myriam Hernandez Alvarez Pablo Marcillo, "Modeling of a Vehicle Accident Prediction System Based on a Correlation of Heterogeneous Sources," in *Advances in Human Aspects of Transportation*., 2020.

[7] Mingze Xu, Yuchen Wang, David Crandall, Ella M Atkins Yu Yao,., 2019.

[8] M. Kurian, R. Joseph, R. Sruthi and S. Thomas M. Philip, "A Computer Vision Approach for Lane Detection and Tracking," in *2nd International Conference on Advances in Computing, Communication, Embedded and Secure Systems (ACCESS)*, 2021.

[9] S. Lin and C. Lee, "Pedestrians and vehicles recognition based on image recognition and laser distance detection," in *16th International Conference on Control, Automation and Systems (ICCAS)*, 2016.

[10] S. Amila, S. Kaushalya, S. Chandrasiri and V. Piyawardana E. Siriwardana, "Driving Through a Bend: Detection of Unsafe Driving Patterns and Prevention of Heavy Good Vehicle Rollovers,"

in *2nd International Informatics and Software Engineering Conference*, 2021.

[11] P. Raizada, A. Kushwaha, R. Bhatnagar and S. Saboo R. Jain, "Blind Vision and Theft Protection Device in Vehicles using Embedded Systems," in *International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*.

[12] E. Ijjina and S. Sharma, "Accident detection from dashboard camera video," in *10th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, 2019.

[13] S. Gupta, I. Kavati D. Chand, "Computer Vision based Accident Detection for Autonomous Vehicles," in *2020 IEEE 17th India Council International Conference (INDICON)*, 2020.

[14] Xun Zuo, Jiaojun Li, Jie Huang, Fan Yang, and Tian Qiu, "Pedestrian detection based on one-stage yolo algorithm," 2021.

[15] Zillur Rahman, Amit Ami, and Muhammad Ullah, "A real-time wrong-way vehicle detection based on yolo and centroid tracking," , 2020.

[16] Ernin Niswatul Ukhwah, Eko Mulyanto Yuniarno, and Yoyon Kusnendar Suprapto, "Asphalt pavement pothole detection using deep learning," in *2019 International Seminar on Intelligent Technology and Its Applications*, 2019.

[17] Mate Kriˇsto, Marina Ivasic-Kos, and Miran Pobar, " Thermal object detection in difficult weather conditions using yolo," *IEEE Access*, 2020.

[18] J.F. Pagel. and Philip Kirshtein, "Neural Networks: The Hard and Software Logic," in *Machine Dreaming and Consciousness*.: Academic Press, 2017, ch. 6, pp. 83-92, DOI: 10.1016/B978-0-12-803720-1.00006-2.

[19] Ed Burns and John Burke. (2021) SearchEnterpriseAI. [Online]. https://www.techtarget.com/searchenterpriseai/definition/neural-network

[20] Shubham Periwal. (2020) Real-Time Object Detection with YOLO.

[21] Manish Chablani. (2017) YOLO — You only look once, real time object detection explained.

[22] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger, "Densely Connected Convolutional Networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, URl: http://arxiv.org/abs/1608.06993.

[23] Prof. L. K. Wani, Md Maaz Momin, Sharwari Bhosale, Abhishek Yadav, and Manas Nil, "Vehicle Crash Detection using YOLO Algorithm," *International Journal of Computer Science*

and Mobile Computing, 2022.

[24] Blue & Green Tomorrow. (2016) Blue & Green Tomorrow. [Online]. https://blueandgreentomorrow.com/environment/traffic-accidents-harm-environment/

[25] World Health Organization. (2022) WHO. [Online]. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries#:~:text=Every%20year%20the%20lives%20of,a%20result%20of%20their%20injury.

[26] Monika Másilková, "Health and social consequences of road traffic accidents," *Kontakt*, vol. 19, pp. e43-e47, 2017, https://doi.org/10.1016/j.kontakt.2017.01.007}.

[27] Md Jamil Ahsan, Sourav Roy, and Armana Sabiha Huq, "AN IN-DEPTH ESTIMATION OF ROAD TRAFFIC ACCIDENT COST IN BANGLADESH," in *International Conference on Sustainable Development in Technology for 4th Industrial Revolution 2021*, 2021.

[28] (2022) HSEWatch. [Online]. https://hsewatch.com/effects-of-road-accidents/

[29] M. Xu, Y. Wang, D. Crandall and E. Atkins Y. Yao, "Unsupervised Traffic Accident Detection in First-Person Videos," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

Codes

The computer program for our system model is written in python using Google Colab libraries.

Detection:

This portion of our code has been used for training with our dataset and testing with the test set to get a result of its performance.

```
from google.colab import drive

drive.mount('/content/drive')

import numpy as np

import torch

import torch.nn.functional as F

import torch.optim as optim

import torch.utils.data as data_utils

import torchvision

from PIL import Image, ImageFile

from torch import nn

from torch import optim as optim

from torch.autograd import Variable

from torch.optim import lr_scheduler

from torch.utils.data import DataLoader, Dataset

from torch.utils.data.sampler import SubsetRandomSampler

from torchvision import datasets, models, transforms

from torchsummary import summary
```

```
train_on_gpu = torch.cuda.is_available ()
if not train_on_gpu:
   print ('CUDA is not available.  Training on CPU ...')
else:
   print ('CUDA is available!  Training on GPU ...')
ImageFile.LOAD_TRUNCATED_IMAGES = True

data_dir = ('/content/drive/MyDrive/cse499_data/')
valid_size = 0.2
train_transforms = transforms.Compose([transforms.RandomRotation(30),
                                       transforms.RandomResizedCrop(224),
                                       transforms.RandomVerticalFlip(),
                                       transforms.RandomHorizontalFlip(),
                                       transforms.ToTensor(),
                                       transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])])
                                       test_transforms = transforms.Compose([transforms.Resize(255),
                                      transforms.CenterCrop(224), transforms.ToTensor()])
valid_transforms = transforms.Compose([transforms.RandomRotation(30),
                                       transforms.RandomResizedCrop(224),
                                       transforms.RandomVerticalFlip(),
                                       transforms.RandomHorizontalFlip(),
                                       transforms.ToTensor(),
                                       transforms.Normalize([0.485, 0.456, 0.406],  [0.229, 0.224, 0.225])
```

```
train_data = datasets.ImageFolder(data_dir, transform=train_transforms)
train_data, test_data, valid_data = torch.utils.data.random_split(train_data, [27781, 5954, 5954])
trainloader = torch.utils.data.DataLoader(train_data, batch_size=8, num_workers=1,
                    pin_memory=True)
testloader = torch.utils.data.DataLoader(test_data, batch_size=8, num_workers=1,
                    pin_memory=True)
validloader = torch.utils.data.DataLoader(valid_data, batch_size=8, num_workers=1,
                    pin_memory=True)
n_classes = 2


def get_num_correct(preds, labels):
    return preds.argmax(dim=1).eq(labels).sum().item()
model = models.densenet161()
model.classifier = nn.Sequential(nn.Linear(2208, 1000), nn.ReLU(), nn.Dropout(0.2),
                            nn.Linear(1000, 2), nn.LogSoftmax(dim=1))
criterion = nn.NLLLoss()
# Only train the classifier parameters, feature parameters are frozen
optimizer = optim.Adam(model.parameters(), lr=0.001)
scheduler = lr_scheduler.StepLR(optimizer, step_size=7, gamma=0.1)
model = model.cuda()
total_params = sum(p.numel() for p in model.parameters())
print(f'{total_params:,} total parameters.')
total_trainable_params = sum(p.numel() for p in model.parameters() if p.requires_grad)
print(f'{total_trainable_params:,} training parameters.')
```

```
total_correct = 0.0
epoches = 100
valid_loss_min = np.Inf
for epoch in range(1, epoches+1):
    torch.cuda.empty_cache()
# keeping track of training and validation loss
    train_loss = 0.0
    valid_loss = 0.0
  model.train()
    for data, target in trainloader:
                if train_on_gpu:
            data, target = data.cuda(), target.cuda()
        # clearing the gradients of all optimized variables
        optimizer.zero_grad()
        # forward passing
        output = model(data)
        # calculating the batch loss
        loss = criterion(output, target)
        # backward passing
        loss.backward()
        # update parameter
        optimizer.step()
        # update training loss
        train_loss += loss.item()*data.size(0)
        scheduler.step()
```

```
        total_correct += get_num_correct(output, target)
model.eval()
for data, target in validloader:
        torch.cuda.empty_cache()
        # moves tensors to GPU if CUDA is available
        if train_on_gpu:
            data, target = data.cuda(), target.cuda()
            # forward passing: predicted output
        output = model(data)
        loss = criterion(output, target)
        valid_loss += loss.item()*data.size(0)
    train_loss = train_loss/len(trainloader.sampler)
    valid_loss = valid_loss/len(validloader.sampler)
    print('Epoch: {} \tTraining Loss: {:.6f} \tValidation Loss: {:.6f}'.format( epoch,
                                                                    train_loss,valid_loss))

    # save if validation loss has decreased
    if valid_loss <= valid_loss_min:
        print('Validation loss decreased ({:.6f} --> {:.6f}).  Saving model ...'.format(valid_loss_min,
                                                                        valid_loss))
        torch.save(model.state_dict(), 'sed.pt')
        valid_loss_min = valid_loss
```

Prediction:

This portion of the code is to take input videos, run it through our model and give a prediction.

```
import webbrowser

import cv2

import numpy as np

import torch

import torch.nn.functional as F

import torch.optim as optim

import torch.utils.data as data_utils

import torchvision

from PIL import Image, ImageFile

from torch import nn

from torch import optim as optim

from torch.autograd import Variable

from torch.optim import lr_scheduler

from torch.utils.data import DataLoader, Dataset

from torch.utils.data.sampler import SubsetRandomSampler

from torchvision import datasets, models,transform

train_on_gpu = torch.cuda.is_available()

if not train_on_gpu:

    print('CUDA is not available.  Training on CPU ...')

else:

    print('CUDA is available!  Training on GPU ...')

ImageFile.LOAD_TRUNCATED_IMAGES = True
```

```
test_transforms = transforms.Compose([transforms.Resize(255),transforms.CenterCrop(224),
                                      transforms.ToTensor()])
model = models.densenet161()
model.classifier = nn.Sequential(nn.Linear(2208, 1000), nn.ReLU(),
nn.Dropout(0.2),nn.Linear(1000, 2), nn.LogSoftmax(dim=1))
criterion = nn.NLLLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)
scheduler = lr_scheduler.StepLR(optimizer, step_size=7, gamma=0.1)
model = model.cuda()
model.load_state_dict(torch.load('tensorboardexp.pt'))
classes = ["accident", "noaccident"]
count = 0
counts = 1
vid = cv2.VideoCapture(0)
ret = True
while ret:
    if ret == True:
        ret, frame = vid.read()
        try:
            img = Image.fromarray(frame)
        except ValueError:
            break
        except AttributeError:
            break
```

```
        img = test_transforms(img)
        img = img.unsqueeze(dim=0)
        img = img.cuda()
        model.eval()
        with torch.no_grad():
            output = model(img)
            predicted = torch.max(output, 1)
            index = int(predicted.item())
            if index == 0:
                cv2.imwrite(r"D:\xamp\htdocs\img\frame%d.png" % count, frame)
                count += 1
                if counts == 1:
                    webbrowser.open('127.0.0.1', new=2)
                    counts += 1
            labels = 'status: ' + classes[index]
        cv2.putText(frame, labels, (10, 100), cv2.FONT_HERSHEY_DUPLEX, 2, (0, 0, 255), 5, cv2.LINE_AA)
        cv2.imshow('Frame', frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break


vid.release()
cv2.destroyAllWindows()
```