Information technology institute

Car collision detection system

January 2022

April 2021

Information technology institute

By:

Mohamed Magdi [Nasr city –Group 8]
Esraa Hazem [Alexandria–Group 1]
Amr Eid [Nasr city –Group 5]
Mohamed sebaie [Nasr city-Group 3]

Under Supervision of:

[Roba Gamal]
[Data scientist],
Orange Innovation Egypt.

[George Iskandar]
[Cluster Head of ITI Ai-Pro Program], Information technology institute.

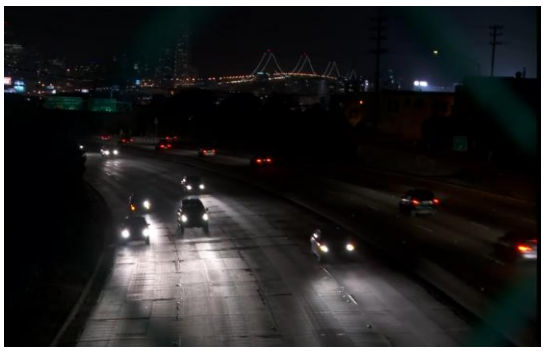
Table of Contents

Acknowledgement

First of all, We would like to thank Allah for being with us in all our ups and downs and for giving us the strength to move forward along this journey of hard work and in life generally, We are grateful to our parents and family who were always providing help and support throughout the whole years of study, We hope we can give that back to them, Also We want to express our gratitude for our supervisor Eng. Roba Gamal, he has always been supportive to us through the whole work, we would like to thank Eng. George Iskandar for all his great efforts, finally we thank all of our fellow students in all groups for this heartwarming, supportive and competitive community that was like a huge family to us.

1. Introduction with literature review Methodology used:

Car Crash can cause property damage, injuries, death, and nonrecurrent congestions. Accurate and fast crash detection can help improve the response speed of incident management, which in turn reduces injuries/fatalities and congestions induced by crash occurrence. Thus, A Car Crash detection system will decrease the response time of emergency services. Ambulances will arrive faster and traffic jams due to the accident will be less harmful. More accurate accident statistics will be collected. Such an algorithm could also be applied on the data which have been recorded on the server, to increase the volume of the information available for the analysis in further training of more complex models using real data.



Unfortunately, car crash detection problem still does not have reliable solutions. The existing algorithms of classical computer vision are based on hand- crafted features, which are adapted for normal lighting condition and therefore work not so well with non-standard lighting. Figure 1 shows a frame with night vision that can make it difficult for an automated system to analyze the video.

Furthermore, accidents usually happen on streets, which are often crowded such. For example, Fig. 2

Even under normal conditions it is not always possible to accurately recognize accidents as collisions can occur in different ways. These are some of the reasons that make automatic road accident detection a difficult task.

Traditional crash/incident detection methods mostly rely on traffic flow modeling techniques. The basic idea of traffic flow modeling is to identify nonrecurrent congestion, based on data from loop detectors, microwaves, and probe. However, nonrecurrent congestion and recurrent congestion can be difficult to be differentiated without enough and sound historical data. Thus, the performance of traffic flow modeling approach heavily depends on the data quality obtained from traffic detectors. Moreover, it could often fail when the traffic environment is too complex (e.g., multimodal traffic in urban area). Thus, detection accuracy of such method is sometimes not guaranteed.

Another emerging method is to identify incident based on crowd sourcing data. However, such method could also suffer from under reporting issues when there is no witness around the incident scene. Nowadays, with the development of intelligent transportation systems (ITS), video cameras have been widely installed in many cities and highways. Thanks to their wide coverage, vision-based crash detection techniques have gained increasing research attention in the recent years. Their basic concept is to automatically identify crash scenes based on the features of traffic images/videos through computer-vision techniques. Such techniques as a promising intelligent crash detection method, are expected to significantly reduce human labors and have achieved relatively high detection accuracy. To ensure detection accuracy, a video-based crash detection method needs to be capable of extracting important crash features from traffic images/videos.

In general, there are two main types of features of interest:

motion (temporal) features and appearance (spatial) features. Appearance features include apparent vehicle damage, vehicle rollovers, and pedestrian fallen-off. Motion features need to be continuously identified, including the intersection of vehicle trajectories and the gathering of pedestrians. From this perspective, current crash detection methods can be classified into two groups: motion feature-based methods and feature fusion-based methods.

Main Types of Features of Interest

Motion (Temporal)

- Motion features need to be continuously identified, including the intersection of vehicle trajectories and the gathering of pedestrians.

Appearance (Spatial)

- Appearance features include apparent vehicle damage, vehicle rollovers, and pedestrian fallen-off.

From This Perspective, Current Crash Detection Methods Can Be Classified into Two Groups:

- **Motion Feature-based Methods**
- **Feature Fusion-based Methods**

Many research works are based on motion features, such as

- The Intersection of Vehicle Trajectories,
- The Overlap of Bounding Box Detectors,
- The Speed Change of Vehicles.

Some Used Background Subtraction Methods to Extract Vehicles' Motion Features (Acceleration, Direction, And Velocity), Based on Which Certain Rules and Thresholds Were Applied to Identify Crashes.

- **Maalou et al.** tracked vehicles' motion based on optical flow methods and used heuristic methods to find a threshold for crash identification.
- **Sadeky et al.** used Histogram of Flow Gradient (HFG) as the motion features and discriminated crash from non-crash, based on logistic regression.
- **Chen et al.** developed an Extreme Learning Machine (ELM) for crash identification, based on motion features represented by Scale-Invariant Feature Transform (SIFT) and optical flow.

In recent years, with the development of deep learning methods (e.g., Faster R-CNN (Faster Region-based CNN) and YOLO (You Only Look Once)), the performance of vehicle detection and tracking has been significantly improved.

- **Vicente and Elian** used YOLO model to detect motion features and used support vector machine (SVM) for crash identification.
- **Arceda and Riveros** propose a car crash detection system that combines a violent flow (ViF) descriptor and SVMs using closed-circuit television (CCTV) video data, which are detected through CNN.
- **Lee and Shin** used Faster R-CNN for vehicle detection and Simple Online and Real-Time tracking (SORT) for vehicle tracking.
- Based on those motion features, the incident/crashes in tunnels were detected.
- **Paul** applied Mask R-CNN (Mask Region-based CNN) for motion feature extraction and used rules for crash detection.

Motion feature-based models only depend on vehicle motions. This requires a high precision of object detection and tracking.

When the traffic environment is complicated, vehicle detection and tracking performance could be decreased, resulting in low crash detection performance. Furthermore, some crashes may not be detected only based on motions, such as vehicle rollover and pedestrian fallen-off.

Recently**, the feature fusion-based** (i.e., appearance and motion) crash detection methods have become increasingly popular.

There are two types:

1. One is based on unsupervised learning methods. For instance, Singh and Mohan [26] and Yao [27] developed a crash detection model based on auto-encoder methods.

2. Another type is based on supervised learning framework, which normally combines**:** a module (e.g., convolutional neural network) for spatial feature extraction and a module (e.g., a recurrent neural network) for temporal feature extraction.

    - **Batanina et al.** used Convolutional 3D (C3D) model to capture both spatial and temporal crash features from simulated video crashes. Then, a domain adaption (DA) transfer learning was applied to the real-world condition. The accuracy has been improved by 10%. **(Future Work)**
    - **Huang et al.** employed two-stream network to separately extract appearance features and motion features, which were then further combined to detect crashes.

According to previous literature, the performance of crash detection can be improved by feature fusion methods.

Although feature fusion-based methods have achieved a better performance than motion feature-based methods, some improvements still can be made.

Previous work

1-Proposed Framework:

Dataset: collected from YouTube
Detection: MASK R-CNN
Tracking: centroid based tracking
techniques of accident based on 3 anomalies:

-----> Vehicle overlaps
-----> Car trajectory (Cosine Arc between trajectories)
-----> Change in velocity

Drawbacks: The tracking of vehicles is a key component in the accident detection but tracking in the dense traffic and abrupt motion is a challenging problem because the scenario typically contains abrupt changes in the appearance and motion of the target

2-Vision based:

Dataset: UNKNOWN
Detection: GMM (Background Subtraction)
Tracking: mean shift
techniques of accident based on 3 anomalies:

-----> position changes
-----> acceleration
-----> direction (uses arctan to detect car direction)

Drawbacks: Not suitable in unconstrained environments like a frequent change in traffic pattern and weather conditions since it relays only on the change in position and speed parameters.

3-Deep spatio-temporal:

Dataset: collected CCTV surveillance camera in Hyderabad India
https://sites.google.com/site/dineshsinghindian/iith_accident-dataset

Course of accident can be divided into three stages:

pre-collision-------> good evidence for crime scene investigation. The pre-collision situation is a clear violation of traffic rules by any/both the vehicles, which include violation of traffic lane, violation of signals at intersections, violation of speed limit at congested roads, abrupt motion on the road.
collision----------> detect a collision is to identify the joints of the trajectories of the vehicles over spatiotemporal dimensions.
post-collision. -----> To check the near environment after the collision if it's a collision or just an occlusion

Challenges the discrimination between collision and occlusion.

Solution: For this we use the trajectories over space-time interest points [39] and improved dense trajectories.

The two most common post-collision scenes include:
1) Fallen objects at the collision point.
2) Crowd attention towards the collision point.
Algorithms:

The anomaly detection works in two steps, the first step is the automatic training of the deep features, and the second step is to determine the outlier

4-Mixed Traffic Flow:

Dataset: collected from many sources (Annotated for YOLO)
Detection: YOLO
techniques of accident detection:

-----> Retinex image enhancement algorithm.
-----> YOLO for object detection.
-----> decision tree-based framework

5- **A New Video-Based Crash Detection Method- Balancing Speed and Accuracy Using a Feature Fusion Deep Learning Framework**

- First, Introduced Attention Module into Residual Neural Networks to Improve the Performance of Detecting Local Appearance Features.
- Meanwhile, We Linked Resnet With CONV- LSTM Model to Simultaneously Capture Crashes' Appearance and Motion Features.

Tried methods:

a) Yolo v3, deep sort, Rule based:

- Yolo v3 was the used method to detect cars. Deep sort was used for tracking.
- Rule based to detect accident dependent on feature extraction:

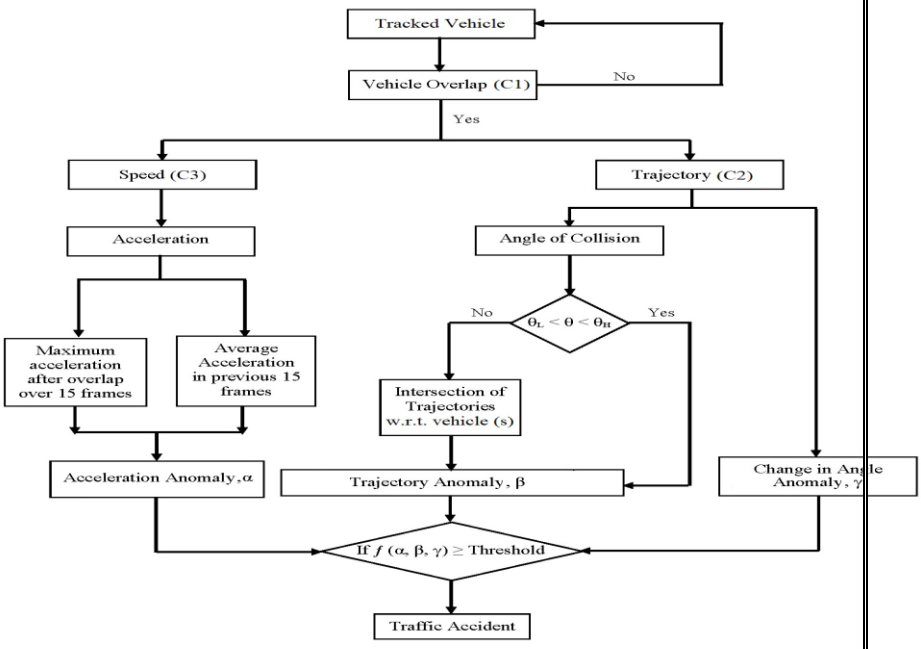
C1: The overlap of bounding boxes of vehicles

C2: Determining Trajectory and their angle of intersection

C3: Determining Speed and their change in acceleration

- Accident Detection:

1) Acceleration Anomaly, α
2) Trajectory Anomaly, β
3) Change in Angle Anomaly, ϒ



When two vehicles are overlapping, we find the acceleration of the vehicles from their speeds captured in the dictionary. We find the average acceleration of the vehicles for 15 frames before the overlapping condition (C1) and the maximum acceleration of the vehicles 15 frames after C1. We find the change in accelerations of the individual vehicles by taking the difference of the maximum acceleration and average acceleration during overlapping condition (C1). The Acceleration Anomaly (α) is defined to detect collision based on this difference from a pre-defined set of conditions. This parameter captures the substantial change in speed during a
collision thereby enabling the detection of accidents from its variation.
The Trajectory Anomaly (β) is determined from the angle of intersection of the trajectories of vehicles (Θ) upon meeting the overlapping condition C1.
1) If Θϵ (Θl, Θh), βis determined from a pre-defined set of conditions on the value of Θ.
2) Else, β is determined from Θ and the distance of the point of intersection of the trajectories from a predefined
set of conditions.
Thirdly, we introduce a new parameter that considers the abnormalities in the orientation of a vehicle during a collision. We determine this parameter by determining the angle (Θ) of a vehicle with respect to its own trajectories over a course of an interval of five frames. Since in an accident, a vehicle undergoes a degree of rotation with respect to an axis, the trajectories then act as the tangential vector with respect to the axis. By taking the change in angles of the trajectories of a vehicle, we can determine this degree of rotation and hence
understand the extent to which the vehicle has underwent an orientation change. Based on this angle for each of the vehicles in question, we determine the Change in Angle Anomaly (ϒ) based on a pre-defined set of conditions.
Lastly, we combine all the individually determined anomaly with the help of a function to determine whether or not an accident has occurred. This function f (α; β; ϒ) considers the weightages of each of the individual thresholds based on their values and generates a score between 0 and 1. A score which is greater than 0:5 is considered as a vehicular accident else it is discarded. This is the key principle for
detecting an accident.

Challenges:

In yolo v3 and deep sort:

- Struggles to detect close objects because each grid can propose only 2 bounding boxes.
- Struggles to detect small objects.
- The detected object could loss the original id and get new id

In Rule based:

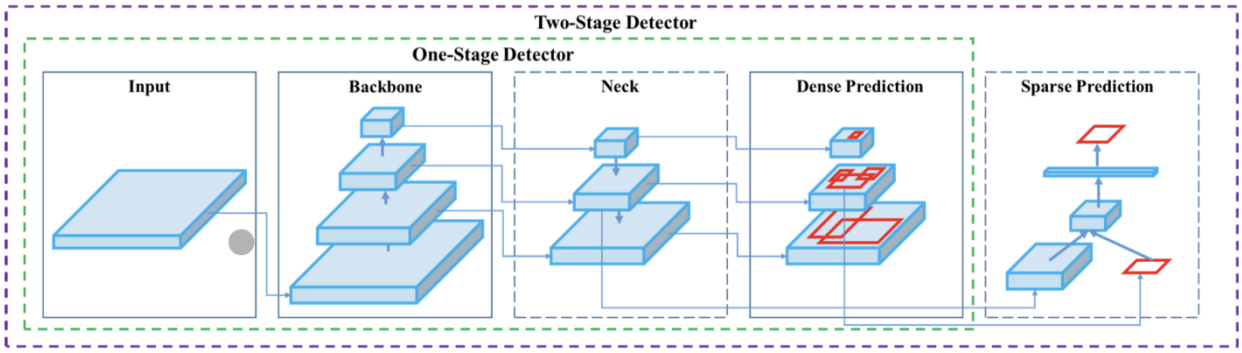
- Each rule needs specific threshold to detect the abnormal event
- Concatenation of the effect of each rule to detect the accident

Violent flow model

1-segmentation:

a-detect cars using Yolo v4:

It is a milestone model which solidified YOLO's name and position in the computer vision field. It was released with the concept of BoF (bag of freebies) and BoS (bag of specials) techniques to enhance model performance.



BoF: Techniques that enhance model accuracy without increasing the inference cost (computation or inference times). Examples: Data augmentation, regularization, Normalization.

BoS: Techniques that improve accuracy while slightly increasing the inference cost. These techniques are generally in the form of plugin modules i.e. One can add or remove these techniques from the model at any time. Examples: Spatial attention modules, non-max suppression, Non-linear activations.

CSPDarknet53: The Cross Stage Partial architecture is derived from the DenseNet architecture which uses the previous input and concatenates it with the current input before moving into the dense layer.

For the purpose of the YOLOv4 object detection tutorial, we will be making use of its pre-trained model weights on Google Colab. The pre-trained model was trained on the MS-COCO dataset which is a dataset of 80 classes engulfing day-to-day objects. This dataset is widely used to establish a benchmark for the purposes of detection and classification. We will showcase its powerful object detection capabilities on both images and videos.

b- Tracking each car (Deep Sort):

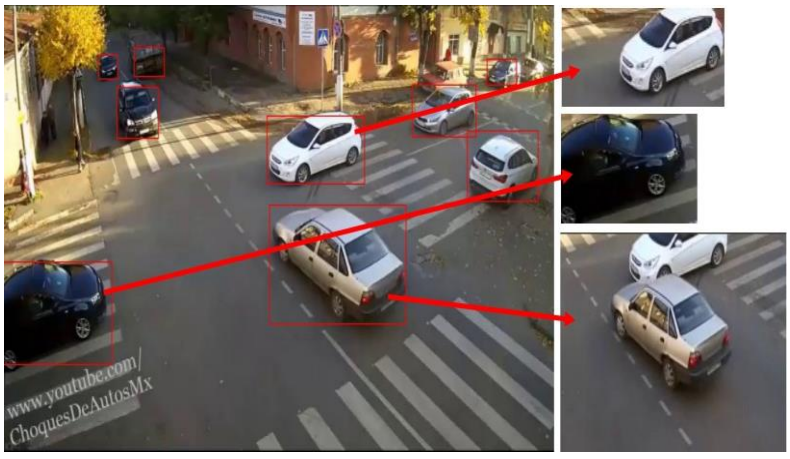
One of the most widely used, object tracking framework is Deep SORT

Kalman Filter

The Kalman filter for tracking moving objects estimates a state vector comprising the parameters of the target, such as position and velocity, based on a dynamic/measurement model. We know different movement conditions and occlusions can hinder the vision tracking of an object. It is considered to use the capacity of the Kalman filter which allow small occlusions and complex movements of objects

C-Making trajectory for each car:

After the tracker we get the boundary box of each detected object. then use these dimensions to crop each object from the frame as shown

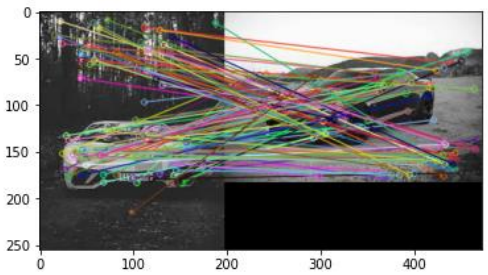


d-Create sequence:

we must save each cropped objects according to the id from the tracker, but we have a challenge in this step as the id of the object can be changed for any sudden change to the object, so we check the similarity features after adding the new cropped object

-similarity (feature extraction from images)" SIFT":

SIFT helps locate the local features in an image, commonly known as the '*key points* 'of the image. These key points are scale & rotation invariant that can be used for various computer vision applications, like image matching, object detection, scene detection



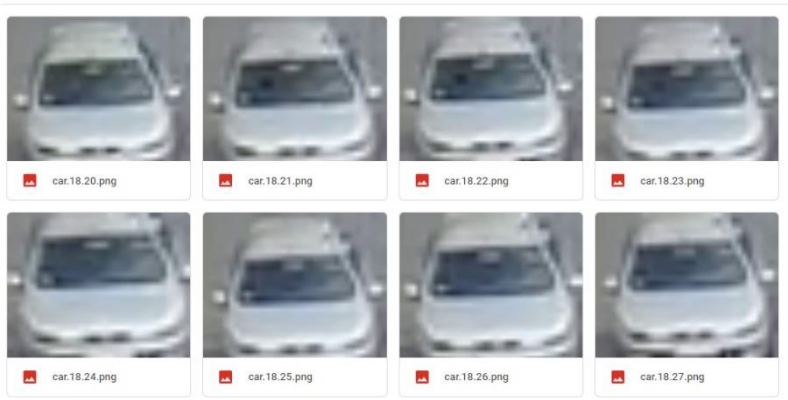
So, we after checking the id of the cropped object we use similarity to compare the cropped object with the latest added cropped object in other objects to check if the cropped object is a new detected object or object has detected before but lost his id!!!!!

-Create video of smaller cars 30 Frame:

- Save the sequence of each object for each 30 frames in a train video
- Label the train videos to accident or not accident object

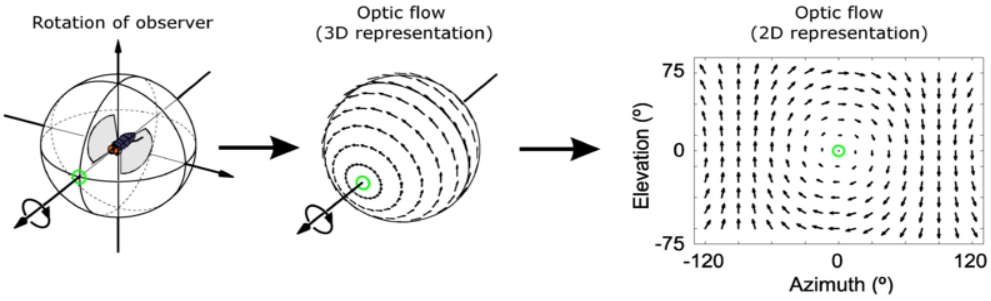
We will use these labeled videos to create dataset to train Violent flow model

After creating dataset, we replace the part of video with a list contain the sequence of 30 frame of each object to be the input in the following step



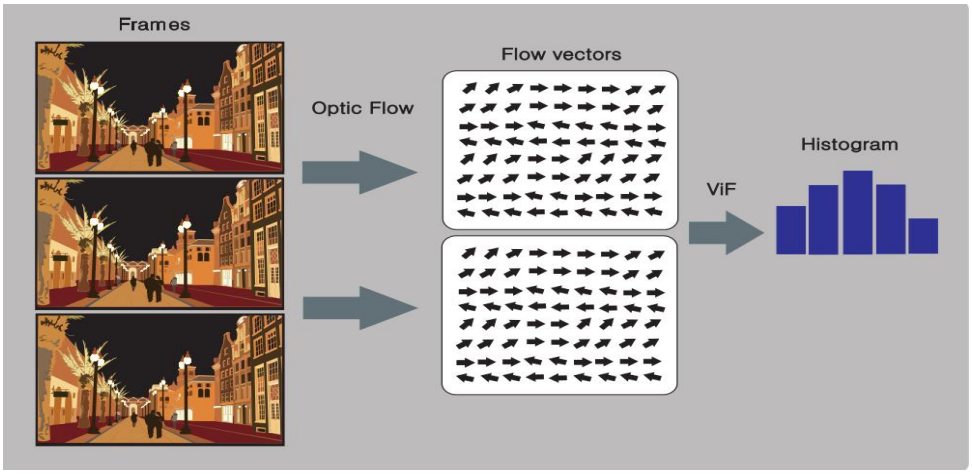
e- Create Violent Flow (VIF):

Optical Flow : Optical flow or optic flow is the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and a scene



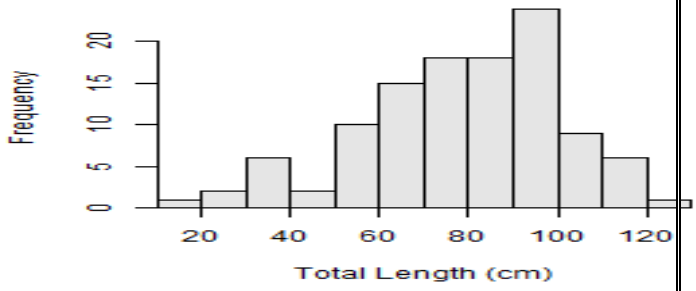
Dense and Sparse Optical Flow: The brief explanation is sparse techniques only need to process some pixels from the whole image, dense techniques process all the pixels. Dense techniques are slower but can be more accurate, but in my experience Lucas-Kanade accuracy might be enough for real-time applications. An example of a dense optical flow algorithm (the most popular) is Gunner FarneBack Optical Flow.

* VIF main Steps:



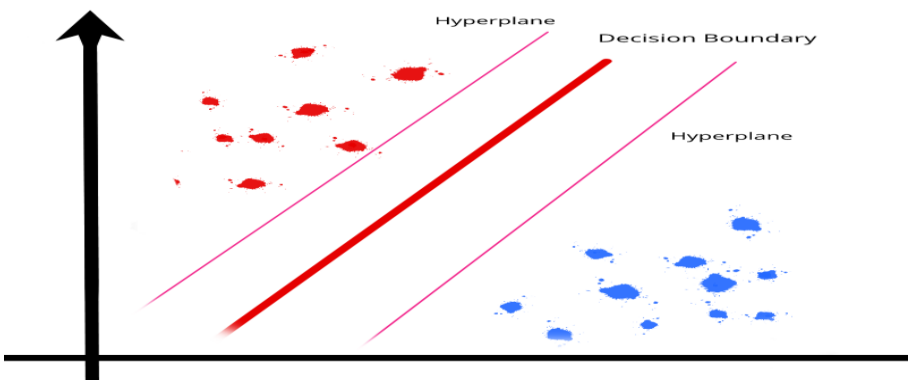
1- Compute FarneBack optical Flow*:* FarneBack optical flow is a dense optical that uses all pixel in the segmented image of object 30 Frames videos and compute angles and magnitude of optical flow vector.

2- Histogram quantization: create histogram on computes features for each frame using 18 bins angle histogram and 18 bins magnitude histogram.



3- Compute VIF: get the mean values on each frame and get only one vector for each 30-frame video the length of feature vector is 36 mixes of angle and magnitude histogram.

4- Train SVM: create datasets contains each object(cars) as a vector and feed svm with these data that is balanced between accident and non-accident objects



3D Convolution

This section provides technical details of our use of CNNs for road accident recognition.

We had a three-fold objective:

First, to build a custom dataset from YouTube Videos,

Second, use this dataset to Find tune Pretrained ResNet3D_18 Pytorch Model which is Pretrained on Kinetics-400 dataset for accident recognition.

Third, Make Threshold for the Model as a calibration step for different areas as a temporary solution for the data shortage to simulate different regions and areas.

CNNs have been shown to provide state-of-the-art results for action recognition as Convolution is an operation of multiplying input data tensor by a convolutional kernel. Based on convolution operation it is possible to build multilayer neural network which can learn how to extract features directly from the data. Such a network is complemented with a densely connected layer with activation function to make class prediction.

The features from these 3D ConvNets encapsulate information related to objects, scenes, and actions in a video, making them useful for various tasks without requiring to finetune the model for each task. C3D has the properties that a good descriptor should have it is generic, compact, simple, and efficient.

Learning Spatiotemporal Features with 3D Convolutional Networks (Facebook AI) 2015

Dataset Task	Sport1M action recognition	UCF101 action recognition	ASLAN action similarity labeling	YUPENN scene classification	UMD scene classification	Object object recognition
Method	[29]	[39]([25])	[31]	[9]	[9]	[32]
Result	**90.8**	75.8 (89.1)	68.7	96.2	77.7	12.0
C3D	85.2	**85.2 (90.4)**	**78.3**	**98.1**	**87.7**	**22.3**

Table 1. **C3D compared to best published results**

Compared to 2D ConvNets, 3D ConvNets has the ability to model temporal information better owing to 3D convolution and 3D pooling operations.

In 3D ConvNets, convolution and pooling operations are performed spatio-temporally while in 2D ConvNets they are done only spatially.

Figure 1 illustrates the difference, 2D convolution applied on an image will output an image, 2D convolution applied on multiple images (treating them as different channels) also results in an image. Hence, 2D ConvNets lose temporal information of the input signal right after every convolution operation.

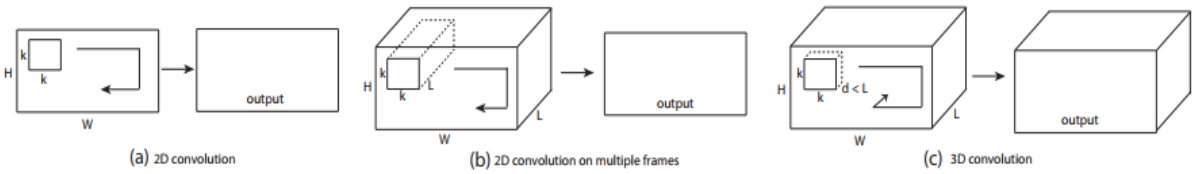


Figure 1. **2D and 3D convolution operations**. a) Applying 2D convolution on an image results in an image. b) Applying 2D convolution on a video volume (multiple frames as multiple channels) also results in an image. c) Applying 3D convolution on a video volume results in another volume, preserving temporal information of the input signal.

Only 3D convolution preserves the temporal information of the input signals resulting in an output volume. The same phenomena are applicable for 2D and 3D polling. Although the temporal stream network takes multiple frames as input, because of the 2D convolutions, after the first convolution layer, temporal information is collapsed completely. Similarly, fusion models in [18] used 2D convolutions, most of the networks lose their input's temporal signal after the first convolution layer. Only the Slow Fusion model (Google Research)2014 uses 3D convolutions and averaging pooling in its first 3 convolution layers. We believe this is the key reason why it performs best among all networks studied. However, it still loses all temporal information after the third convolution layer.

Notations: For simplicity, from now on we refer video clips with a size of c × l × h × w where c is the number of channels, l is length in number of frames, h and we are the height and width of the frame, respectively. We also refer 3D convolution and pooling kernel size by d×k ×k, where d is kernel temporal depth and k is kernel spatial size.

Common network settings: In this section we describe the network settings that are common to all the networks we trained. The networks are set up to take video clips as inputs and predict the class labels which belong to 101 different actions. All video frames are resized into 128 × 171. This is roughly half resolution of the UCF101 frames. Videos are split into non-overlapped 16-frame clips which are then used as input to the networks. The input dimensions are 3 × 16 × 128 × 171. We also use jittering by using random crops with a size of 3 × 16 × 112 × 112 of the input clips during training.

What does C3D learn? We observe that C3D starts by focusing on appearance in the first few frames and tracks the salient motion in the subsequent frames.

Figure 4 visualizes deconvolution of two C3D conv5b feature maps with highest activations projected back to the image space.

In the first example, the feature focuses on the whole person and then tracks the motion of the pole vault performance over the rest of the frames.
Similarly in the second example it first focuses on the eyes and then tracks the motion happening around the eyes while applying the makeup.
Thus, C3D differs from standard 2D ConvNets in that it selectively attends to both motion and appearance. We provide more visualizations in the supplementary material to give a better insight about the learned feature.

Figure 4. **Visualization of C3D model, using the method from [46].** Interestingly, C3D captures appearance for the first few frames but thereafter only attends to salient motion. Best viewed on a color screen.

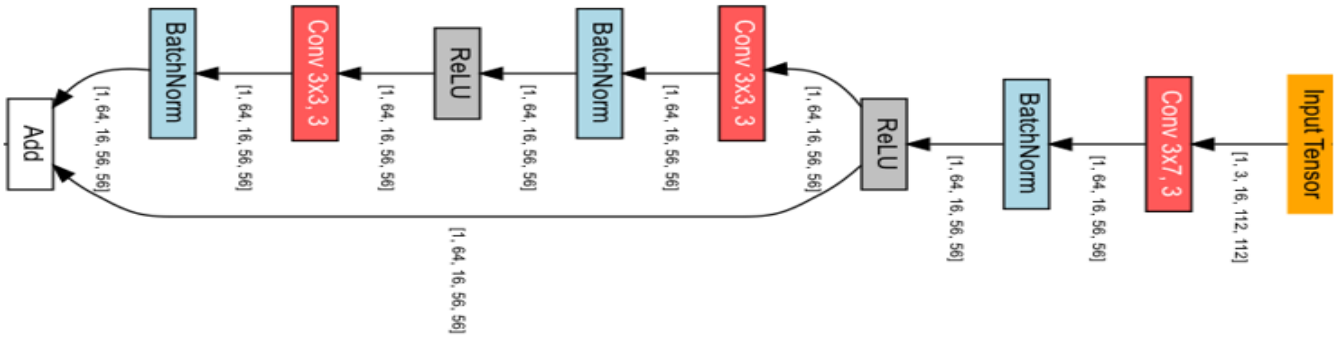
Compared with Recurrent Neural Networks (RNN) based methods, C3D outperforms Long-term Recurrent Convolutional Networks (LRCN) and LSTM composite model by 14.1% and 9.4%, respectively. C3D with only RGB input still outperforms these two RNN-based methods when they used both optical flows and RGB as well as the temporal stream network.

ResNet 3D

It is a type of model for video that employs 3D convolutions. This model collection consists of two main variants.

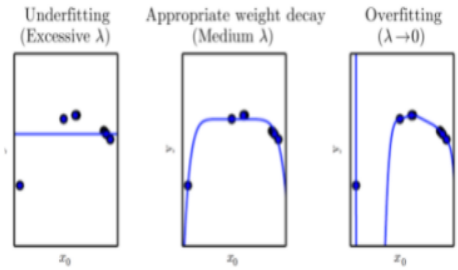
The first formulation is named mixed convolution (MC) and consists in employing 3D convolutions only in the early layers of the network, with 2D convolutions in the top layers. The rationale behind this design is that motion modeling is a low/mid-level operation that can be implemented via 3D convolutions in the early layers of a network, and spatial reasoning over these mid-level motion features (implemented by 2D convolutions in the top layers) leads to accurate action recognition. We show that MC ResNets yield roughly a 3-4% gain in clip-level accuracy over 2D ResNets of comparable capacity and they match the performance of 3D ResNets, which have 3 times as many parameters.

The second spatiotemporal variant is a "(2+1) D" convolutional block, which explicitly factorizes 3D convolution into two separate and successive operations, a 2D spatial convolution and a 1D temporal convolution. The first advantage is an additional nonlinear rectification between these two operations. This effectively doubles the number of nonlinearities compared to a network using full 3D convolutions for the same number of parameters, thus rendering the model capable of representing more complex functions. The second potential benefit is that the decomposition facilitates the optimization, yielding in practice both a lower training loss and a lower testing loss.



Training Techniques	Weight Decay, SGD with Momentum
Architecture	1x1 Convolution, Bottleneck Residual Block, Batch Normalization, 3D Convolution, Global Average Pooling, Residual Block, Residual Connection, ReLU, Max Pooling, SoftMax
ID	r3d_18
LR	0.01
Epochs	45
LR Gamma	0.1
Momentum	0.9
Batch Size	16
Weight Decay	0.0001
LR Warmup Epochs	10

Weight Decay



Weight Decay, or **L2 Regularization**, is a regularization technique applied to the weights of a neural network. We minimize a loss function compromising both the primary loss function and a penalty on the L2 Norm of the weights:

$$L_{new}(w) = L_{original}(w) + \lambda w^T w$$

where λ is a value determining the strength of the penalty (encouraging smaller weights).

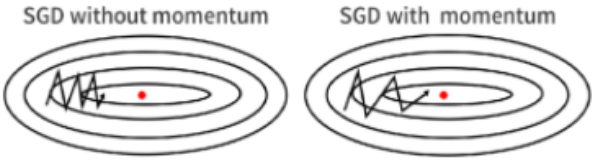
Weight decay can be incorporated directly into the weight update rule, rather than just implicitly by defining it through to objective function. Often weight decay refers to the implementation where we specify it directly in the weight update rule (whereas L2 regularization is usually the implementation which is specified in the objective function).

SGD with Momentum

SGD with Momentum is a stochastic optimization method that adds a momentum term to regular stochastic gradient descent:

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta J(\theta)$$

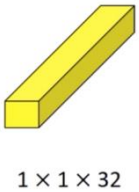
$$\theta_t = \theta_{t-1} - v_t$$



A typical value for γ is 0.9. The momentum name comes from an analogy to physics, such as ball accelerating down a slope. In the case of weight updates, we can think of the weights as a particle traveling through parameter space which incurs acceleration from the gradient of the loss.

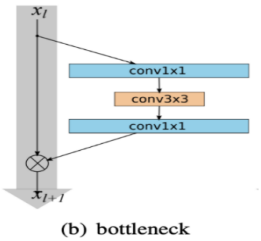
1x1 Convolution

A **1 x 1 Convolution** is a convolution with some special properties in that it can be used for dimensionality reduction, efficient low dimensional embeddings, and applying non-linearity after convolutions. It maps an input pixel with all its channels to an output pixel which can be squeezed to a desired output depth. It can be viewed as an MLP looking at a particular pixel location.



Bottleneck Residual Block

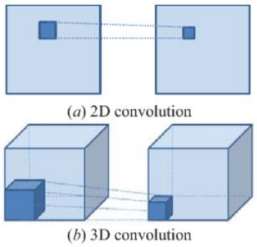
A **Bottleneck Residual Block** is a variant of the residual block that utilizes 1x1 convolutions to create a bottleneck. The use of a bottleneck reduces the number of parameters and matrix multiplications. The idea is to make residual blocks as thin as possible to increase depth and have less parameters. They were introduced as part of the ResNet architecture and are used as part of deeper ResNets such as ResNet-50 and ResNet-101.



(b) bottleneck

3D Convolution

A **3D Convolution** is a type of convolution where the kernel slides in 3 dimensions as opposed to 2 dimensions with 2D convolutions. One example use case is medical imaging where a model is constructed using 3D image slices. Additionally, video-based data has an additional temporal dimension over images making it suitable for this module.


(a) 2D convolution

(b) 3D convolution

Batch Normalization

Batch Normalization aims to reduce internal covariate shift, and in doing so aims to accelerate the training of deep neural nets. It accomplishes this via a normalization step that fixes the means and variances of layer inputs. Batch Normalization also has a beneficial effect on the gradient flow through the network, by reducing the dependence of gradients on the scale of the parameters or of their initial values. This allows for use of much higher learning rates without the risk of divergence. Furthermore, batch normalization regularizes the model and reduces the need for Dropout.

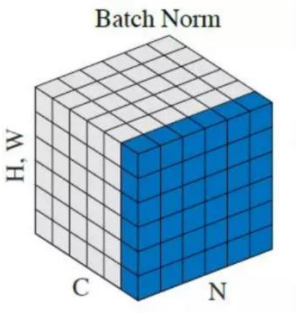
We apply a batch normalization layer as follows for a minibatch B:

$$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} x_i$$

$$\sigma^2_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_{\mathcal{B}})^2$$

$$\hat{x}_i = \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}}$$

$$y_i = \gamma \hat{x}_i + \beta = \mathrm{BN}_{\gamma,\beta}(x_i)$$



Where γ and β are learnable parameters.

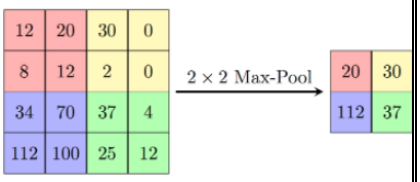
Global Average Pooling

Global Average Pooling is a pooling operation designed to replace fully connected layers in classical CNNs. The idea is to generate one feature map for each corresponding category of the classification task in the last mlpconv layer. Instead of adding fully connected layers on top of the feature maps, we take the average of each feature map, and the resulting vector is fed directly into the SoftMax layer.

One advantage of global average pooling over the fully connected layers is that it is more native to the convolution structure by enforcing correspondences between feature maps and categories. Thus, the feature maps can be easily interpreted as categories confidence maps. Another advantage is that there is no parameter to optimize in the global average pooling thus overfitting is avoided at this layer. Furthermore, global average pooling sums out the spatial information, thus it is more robust to spatial translations of the input.

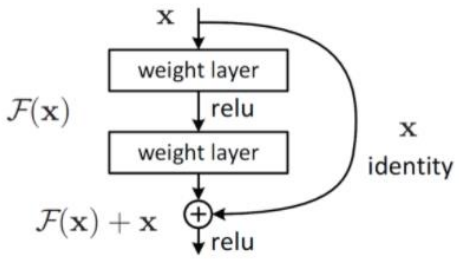
Max Pooling

Max Pooling is a pooling operation that calculates the maximum value for patches of a feature map and uses it to create a down sampled (pooled) feature map. It is usually used after a convolutional layer. It adds a small amount of translation invariance - meaning translating the image by a small amount does not significantly affect the values of most pooled outputs.



Residual Block

Residual Blocks are skip-connection blocks that learn residual functions with reference to the layer inputs, instead of learning unreferenced functions. They were introduced as part of the ResNet architecture.



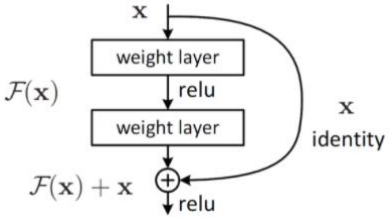
Formally, denoting the desired underlying mapping as H(x), we let the stacked nonlinear layers fit another mapping of F(x): =H(x)−x. The original mapping is recast into F(x)+x. The additional x acts like a residual, hence the name 'residual block'.

The intuition is that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers. Having skip connections allows the network to more easily learn identity-like mappings.

Note that in practice, Bottleneck Residual Blocks are used for deeper ResNets, such as ResNet-50 and ResNet-101, as these bottleneck blocks are less computationally intensive.

Residual Connection

Residual Connections are a type of skip-connection that learn residual functions with reference to the layer inputs, instead of learning unreferenced functions.

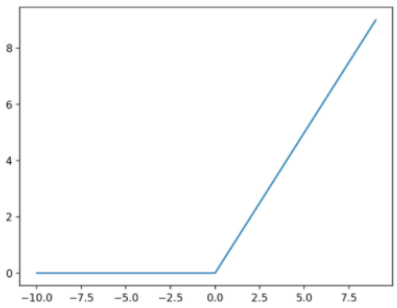


Formally, denoting the desired underlying mapping as H(x), we let the stacked nonlinear layers fit another mapping of F(x): =H(x)−x. The original mapping is recast into F(x)+x.

The intuition is that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers.

Rectified Linear Units

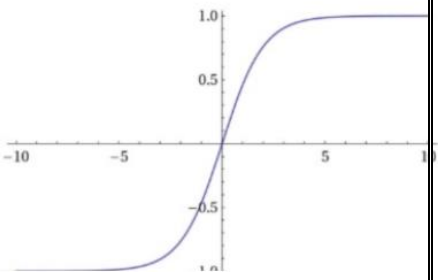
Rectified Linear Units, or **ReLUs**, are a type of activation function that are linear in the positive dimension, but zero in the negative dimension. The kink in the function is the source of the non-linearity. Linearity in the positive dimension has the attractive property that it prevents non-saturation of gradients (contrast with sigmoid activations), although for half of the real line its gradient is zero.



$$f(x) = \max(0, x)$$

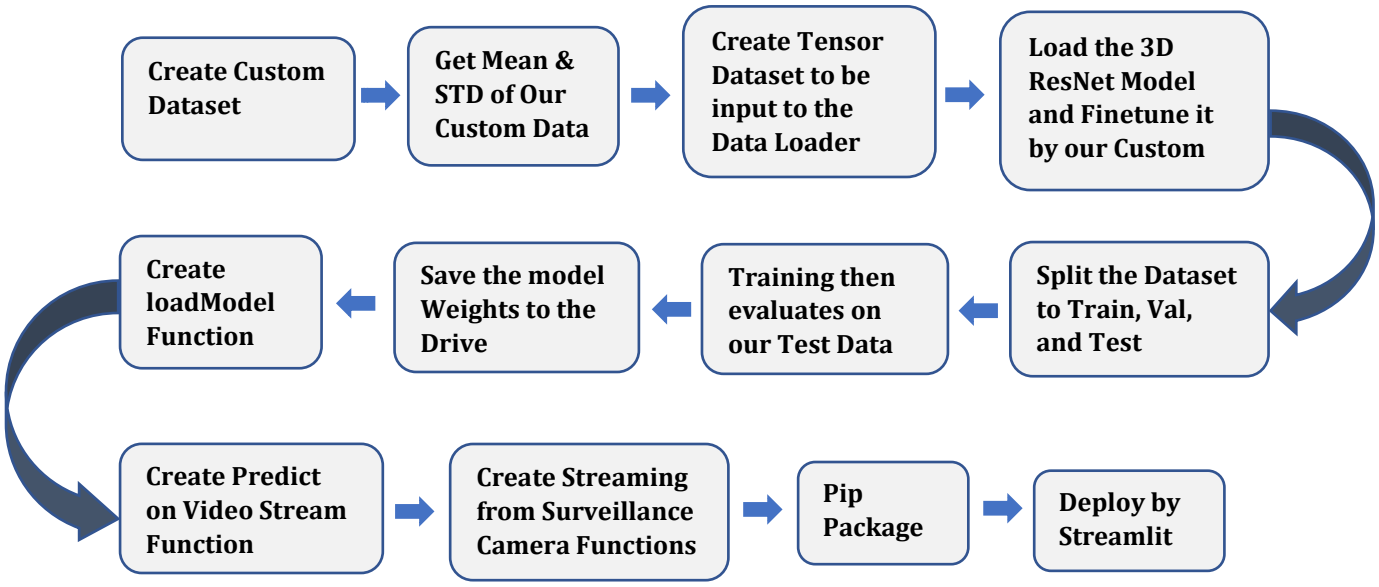
SoftMax

The **SoftMax** output function transforms a previous layer's output into a vector of probabilities. It is commonly used for multiclass classification. Given an input vector x and a weighting vector w we have:



$$P(y = j \mid x) = \frac{e^{x^T w_j}}{\sum_{k=1}^{K} e^{x^T wk}}$$

3D Convolution Pipeline



1- Create Custom Dataset
- As we cut around 20 videos to 98 video each one 15 frames that describe the event (accident or not) 49 Accident and 49 Non_Accident Videos

2- Get Mean & STD
- Get the mean and standard Deviation of the 98 videos to apply in the transform function of Pytorch

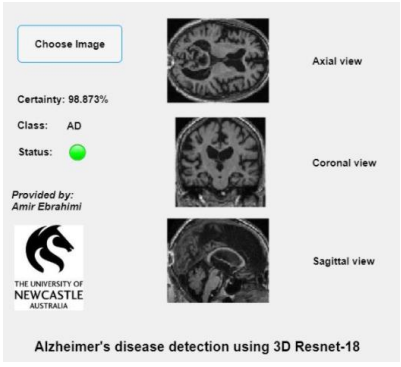
3- Create Dataset to be input to the Data Loader
- Call Create dataset function that take the videos with labels and return two tensors, one for features(x) and the another for labels (y) each one is 5 dimensions (98, 3,15,112,112)

4- Load the 3D ResNet Model and Finetune it by our data
- Load different models form Pytorch video pretrained models

5- Split the Dataset to Train, Val, and Test
- Split the data to 80% Train, 10% Validation, 10 Test

6- Training then evaluates on our Test Data
- Evaluate the data on 23 video and get %

7- Save the model Weights to the Drive
- Save the weights of the best model to drive and r3d_18 is the best one



8- Create load Model Function
9- Create Predict on Video Stream
10- Create Streaming Functions from Cam
11- Pip Package
12- Deploy by Streamlit

Results:

Here's some of related papers results on testing,

The detection rate (DR) I the precision of the model to detect accidents,

The False Alarm Rate (Far) = 1-recall of the accident class

These two metrics are used to measure the performance of car collision detection in papers.

These papers use the total frame and different methodology to detect the accident ex: to create trajectory of detected cars and then use these trajectories to detect overlapping and change in speed anomalies and direction of each car using different techniques,

Our methodology: is evaluated using the same way it's trained using each segmented object from the original frame.

Approach	DR %	FAR %
MASK R-CNN	71	0.53
Vision based	50	0
Deep spatio-temporal	77.53	22.5
Mixed Traffic Flow	92.5	7.5
our First Method Violent flow (SVM)	**62**	**17**
our Second Method 3D convolution (3D ResNet_18)	**92**	**0**

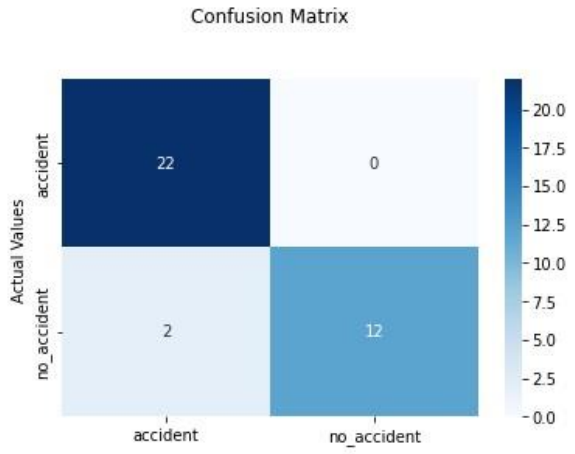
- **Classification Report on test data of**

VIF

```
print(classification_report(results_test,y_test))
```

	precision	recall	f1-score	support
0.0	0.91	0.77	0.83	13
1.0	0.62	0.83	0.71	6
accuracy			0.79	19
macro avg	0.77	0.80	0.77	19
weighted avg	0.82	0.79	0.80	19

3D Convolution



	precision	recall	f1-score	support
accident	0.92	1.00	0.96	22
no_accident	1.00	0.86	0.92	14
accuracy			0.94	36
macro avg	0.96	0.93	0.94	36
weighted avg	0.95	0.94	0.94	36