A Transfer-Learning-Based Approach for Emergency Vehicle Detection

Abubakar M. Ashir[1]

[1]Department of Computer Engineering, Faculty of Engineering, Tishk International University, Erbil, Iraq
Correspondence: Abubakar M. Ashir, Tishk International University, Erbil, Iraq.
Email: abubakar.ashir@tiu.edu.iq

Doi: 10.23918/eajse.v8i1p75

Abstract: The paper presents a computer-vision based approach for real-time detection of different types of emergency vehicles under heavy traffic conditions. This enables preferential path clearance for emergency vehicles by the traffic controller which has the potential of saving lives, properties and increasing the ability to prevent crimes and drastically reducing the total time required by an emergency vehicle to reach its target destination. The main challenge emergency vehicles faced in and around the cities is heavy traffic jams, which significantly hampers their operations resulting in a disastrous outcome. In most of the cities, emergency vehicles are equipped with unique colors and sound system which enable the traffic police to identify them. As our cities become smarter and transition into an era of artificial intelligence, the old system may not be sustainable due to that fact that it needs humans to constantly monitor emergency vehicle arrival at the intersections and also the sound produces by such vehicles may be nuisance and discomforting to the general public. This paper proposed a method of automatic detection of four different categories of emergency vehicle irrespective of the vehicle's shapes, models or manufacturers' using modified version of YOLOv5 object detection algorithm. YOLO is an acronym for (You Only Look Once) and it is an object detection algorithm that divides images into a grid system. Each cell within the grid is responsible for detecting objects within itself. YOLO is one of the most famous object detection algorithms due to its speed and accuracy. YOLO models are used for object detection with high performance which consists of 84 classes to detect and differentiate between 84 different objects. The proposed model developed here is based on 4 classes which are (Firetrucks, Ambulance, Police Car, and Normal Cars) classes. The top layers (fully connected layers) of the YOLO algorithm were re-designed and retrained to get new learned weights while freezing the bottom layers (convolutional layers) and retaining the pre-trained YOLOv5 weights. After retraining with the proposed modified YOLOv5, the model has shown promising results and quite impressive metrics in detecting and classifying emergency vehicles and normal vehicles. Using *Mean Average Precision (mAP*) metric, for police cars we achieved 98%, 96% for fire trucks, 89% for ambulances and 97% for normal cars.

Keywords: Emergency Vehicle, YOLOv5, Deep Learning, Object Detection, Transfer-Learning

1. Introduction

Emergency vehicles such as ambulances, fire trucks, and police cars, provide emergency response to help save lives and ensure security in our society. During emergency conditions, the response time of the emergency vehicle is of significant importance and, in some cases, a few seconds of timesaving may save lives. For example, in the case of an ambulance, each minute of delay can reduce the survival rate of patients with cardiac arrest by 7-10% (Huang et al., 2015). If the ambulance arrives after 8-10 minutes, the chance of survival for those patients will be very low. The importance of response time for firefighters and police cars is needless to explain. A well-populated region like Kurdistan faces too

Received: March 27, 2022
Accepted: May 26, 2022
Ashir, A.M. (2022). A Transfer-Learning-Based Approach for Emergency Vehicle Detection. *Eurasian Journal of Science and Engineering*, *8*(1), 75-89.

many traffic jams. Most of the times emergency vehicles (Ambulance, Firetrucks, etc.) get stuck in traffic causing threats to live.

It's quite significant to prioritize these vehicles and to aid to clear its path, but often it is hard or nearly impossible for traffic police to handle these situations and the rapid increase in the number of cars in cities makes it even much worse. Therefore, an emergency vehicle detection system equipped with vision system (Cameras) will be really useful to reduce traffic jams and help to save lives and properties. Emergency vehicles detection and control in a densely populated city could be one of the important components of Automatic Traffic control and will play an important role in life-threatening situations. Automatic traffic control systems can enhance the response time of emergency vehicles by giving preference to them over others. An effective strategy for realizing this objective is emergency vehicle pre-emption (EVP), which controls the traffic lights in a way that the emergency vehicles can reach their destination in a safer and faster way. To that end, EVP systems make use of different sensing mechanisms to detect the presence of emergency vehicles in the vicinity of traffic lights. If an emergency vehicle is detected, the EVP system will alter the state of traffic signals such that emergency vehicles can pass intersections safely and quickly (Karpis, 2012).

There are some existing solutions proposed by other researchers, which can be classified into in two: those that rely on the sound analysis produce by the emergency vehicle (Sun et al., 2021; Tran & Tsai, 2020) and those that rely on the visual analysis of the vehicles arriving at the intersection (Roy & Rahman, 2019). Mostly the visual approaches use machine learning or deep learning algorithm to implement a detection system based on the visual image input from of the scene. A deep convolutional neural network, which is a type of deep learning, is the most advanced technology nowadays for image classification and detection (Roy & Rahman, 2019). The convolutional neural network uses the concept of a kernel to process the image. The weights of these kernels are learned during training. A deep convolutional neural network is constructed stacking many of such convolutional layers which takes longer time to train because of the need for substantially huge amount of for training and validation (Karpis, 2012). Some of the most prominent state-of-the-art models for detection objects using convolutional deep learning includes Region-based Convolutional Neural Network (R-CNN), Fast-RCNN, Faster R-CNN, Single Shot detector SSD, Deformable Part Model DPM and You-Look-Only-Once YOLO models.

2. Literature and Theoretical Background

2.1 Literature

With the astronomical increase in population density, mostly in urban areas, congestion and difficulties in navigating through such places increases substantially. Accessibility to critical infrastructures and services such as hospitals, security and safety are of paramount importance. Recently, number of researches have surfaced aiming to provide an integrated automatic means of identification of emergency vehicles such as firetrucks, ambulances and security vehicles into traffic control systems.

In a submission titled "Emergency Vehicle Detection on Heavy Traffic Road from CCTV Footage Using Deep Convolutional Neural Network", the authors proposed a method for real-time emergency vehicle detections with a deep convolutional neural network. They reported to have experimented few different pre-trained model convolutional neural networks variants such as VGG-16 (Simonyan & Zisserman, 2014), inception-v3 (Szegedy et al., 2015) and Inception Networks for feature extraction. The detection phase was implemented using a pretrained Yolo-V3 object detector. Yolo-V3 uses a 53-

layer neural network inspired by googLeNet and was trained on ImageNet for classification combined with detection layers. It was implemented on a darknet framework in C language and used CUDA for distributed training on the GPU. For vehicle detection they used YOLO V3 which was pretrained on COCO Common Object in Context (COCO) dataset. For emergency vehicle classification, they used 2 classes (emergency as regular vehicles) as against the 80 classes found in the pretrained model (Roy & Rahman, 2019).

Nellore et al. (2016) calculated the distance of an emergency vehicle using the camera and informing the Traffic Management Center (TMC). They used visual sensing technique. A camera was used to record 1920 × 1080 pixels' video with 30 frames per second. First, the images are converted RGB to grayscale then threshold the images. Some morphological operations were applied to each image and after that, they measured the distance between the camera and the emergency car in a different technique like Euclidean distance in MATLAB. Then they sent the distance, speed and count variable data to the traffic management center to control the traffic signal more effectively for an emergency vehicle. On the other hand, Djahel et al. (2015) designed an advanced traffic control system that minimized the emergency vehicle congestion level. A traffic management controller architecture was made with the help of fuzzy logic controller for emergency services. Their method had got the control of changing the traffic light, changing the speed limit, lane clearance etc. Fuzzy logic determinates most accurate evaluation of the low, medium, high congestion level.

Ondrej Karpis (2012) proposed System for Vehicles Classification and Emergency Vehicles Detection by using a Magnetometers to measure and records all the parameters needed for monitoring traffic data such count, speed, classification, occupancy and presence. They used a sensor that is equipped with magnetometer and microphone to detect emergency vehicles by analyzing the acoustic signals in real-time. The use 256-point Fast Fourier Transform (FFT) with 8 kHz and frequency sampling frequency for recording sounds. Based on the sound frequency range and energy level in FFT, they decide if the vehicle is an emergency one or not. In a similar work titled "Acoustic-Based Emergency Vehicle Detection Using Convolutional Neural Networks", Van-Thuan Tran et al. (2020) proposed an automatic detection system that determines whether there are siren sounds from emergency vehicles nearby to alert other vehicles' drivers to pay attention. They designed a convolutional neural network (CNN)-based ensemble model called SirenNet to classify vehicular sounds into siren sounds, vehicle horns, or noise. They used two streams for processing the sounds in which the stream 1 processes the raw waveform, while the second stream is combined with the extracted feature using techniques known as Mel-frequency cepstral coefficients and log-Mel spectrogram. They reported promising detection accuracy of 98.24% in the siren sound detection tested on different dataset.

Sun et al. (2021) presented a novel system from collecting the real-world siren data to the deployment of models using only two cost-efficient microphones. We are able to achieve promising performance for each task separately, especially within the crucial 10m to 50m distance range to react (the size of our ego vehicle is around 5m in length and 2m in width). The recall rate to determine the existence of sirens is 99.16%, the median and mean angle absolute error is 9.64° and 19.18° respectively, and the median and mean distance absolute error of 9.30m and 10.58m respectively within that range. We also benchmark various machine learning approaches that can determine the siren existence and sound source localization which includes direction and distance simultaneously within 50ms of latency. Authors in (Humayun et al., 2022) proposed system for Emergence Vehicles EV and minimally affects the travel times of other vehicles on the junction. In the presence of an EV in the communication range, the proposed system prioritizes the EV by creating space for it in the lane adjacent to the

shoulder lane. The shoulder lane is a lane that cyclists and motorcyclists will use in normal situations. However, when an EV enters the communication range, traffic from the adjacent lane will move to the shoulder lane. As the number of vehicles on the road increases rapidly, crossing the EV in the shortest possible time is crucial. The EVMS and algorithms are presented in this study to find the optimal vehicle sequence that gives EVs the highest priority. The proposed solution uses cutting-edge technologies (IoT Sensors, GPS, 5G, and Cloud computing) to collect and pass EVs' information to the Roadside Units (RSU). The proposed solution was evaluated through mathematical modeling. The results show that the EVMS can reduce the travel times of EVs significantly without causing any performance degradation of normal vehicles (Humayun et al., 2022).

2.2 YOLO Detector

YOLOv5 or Yolo version 5 is a decedent of the original YOLO unified object detection algorithm published in 2016 by Joseph Redmon (2015), at the Conference on Computer Vision and Pattern Recognition (CVPR). It is very fast and the base model can process up to 45 frames per second which is convenient for real-time object detection. Subsequently in 2017 and 2018, the original author of YOLO released two more versions: YOLOV2 (Redmon & Farhadi, 2016)and YOLOV3 (Redmon & Farhadi, 2018) respectively which are streamlined version of the original YOLO and increase the speed up to 155 frames per second and also introduced the flexibility to tradeoff between accuracy and speed. In 2020 after Joseph Redmon abandoned the development of YOLO, Alexey Bochkovskiy et al. introduced 'YOLOv4 in a paper tilled Optimal Speed and Accuracy of Object Detection' (Bochkovskiy et al., 2020). Like the previous version, the base architecture is retained and uses of state of art BoF (bag of freebies) and several BoS (bag of specials) to improve the accuracy of the detector, without increasing the inference time. While the previous versions of YOLO detectors were all implemented in C language using the Darknet framework, In the same year 2020, Glenn Jocher introduced YOLO v5 (2021)which is an improved of YOLOV4 but implemented using PyTorch libraries in python.

2.3 The Baseline YOLO

The original base architecture of the YOLO (Redmon et al., 2015) provides a unified end-to-end framework for object detection at once unlike its counterparts like R-CNN families (Girshick, 2015; Girshick et al., 2013; Ren et al., 2015), and DPM (Girshick et al. 2014). The algorithm divides an input image into $s \times s$ grid. Each grid cell is responsible for detecting an object whose center falls within it. Similarly, each grid cell will predict B number of bounding boxes and the confidence scores for those boxes. The confidence scores tell how certain or otherwise is the cell that a box contains an object and how accurate is the box is in enclosing the object. Each grid cell also predicts one set of C conditional class probabilities irrespective of the number of bounding boxes B predict by the cell. The class probability tells how confident the cell is about the class of the object within that cell. At the test time the conditional class probabilities are multiplied with individual box confidence scores to get class-specific confidence scores for each box. Therefore, YOLO performs regression using deep learning architecture on the input image. The output of the algorithm is a tensor of size $S \times S \times (5 * B + C)$ as depicted in Figure 1.

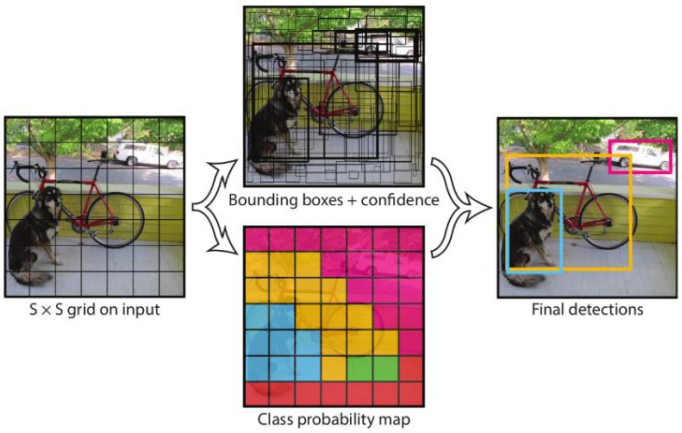


Figure 1: YOLO object detection process

The architecture of the base network is a convolutional neural network inspired by GoogLeNet. They replaced the inception layers in GoogLeNet by 1×1 reduction layers and then 3×3 convolution layers. The model has 24 convolutional layers followed by two fully connected layers as shown in Figure 2 (Redmon et al., 2015).

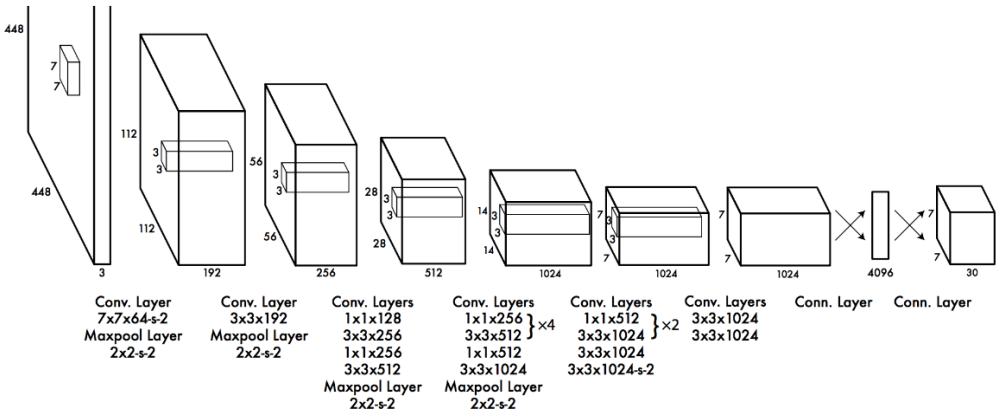


Figure 2: YOLO convolutional and detection layers' architecture

2.4 YOLOv5

YOLO version 5 or YOLOv5 is a direct improvement on YOLOv5 implemented with pyTorch in python instead of C darknet framework used by YOLOv5 (Glenn Jocher, 2021). Like its immediate predecessor, YOLOv5 uses the original baseline YOLO detection architecture but with addition state-of-the-art optimization techniques for convolutional neural networks. These techniques include auto learning bounding box anchors, mosaic data augmentation, the cross-stage partial network, and so on which immensely improve different stages of the base YOLO architecture. YOLOv5 has four important parts just like in YOLOv4 as shown in Figure 4. These parts are the input, backbone, neck, and output. The input part performs the input data preprocessing such as adaptive image filling, mosaic data augmentation and adaptive anchor frame (Li et al., 2022). The adaptive anchor frame automatically set the initial anchor frame size when the dataset changes. The backbone network mainly uses a cross-stage partial network (CSP) and spatial pyramid pooling (SPP) to extract feature maps of different sizes from the input image by multiple convolution and pooling operations. It used BottleneckCSP to reduce the computational cost and accelerate inference, while the SPP extract

features at different scales (up to three-scale feature maps) for the same feature map, which helps improve the detection accuracy. In the neck network, the feature pyramid structures of FPN and PAN are used. The FPN network transmit strong semantic features from the top feature maps into the lower feature maps while the PAN is responsible for transmitting strong localization features from lower feature maps into higher feature maps. The two operations are responsible for strengthening the extracted features from different network layers in Backbone section. The final stage, head, is responsible for predict targets of different sizes on feature maps which uses the backbone YOLO detection approach (Bochkovskiy et al., 2020; Glenn Jocher, 2021; Li et al., 2022). The YOLOv5 consists of four architectures, named YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x. The main difference between them lies in the number of feature extraction modules and convolution kernels at specific locations on the network. The network structure of YOLOv5 is shown in Figure 3.

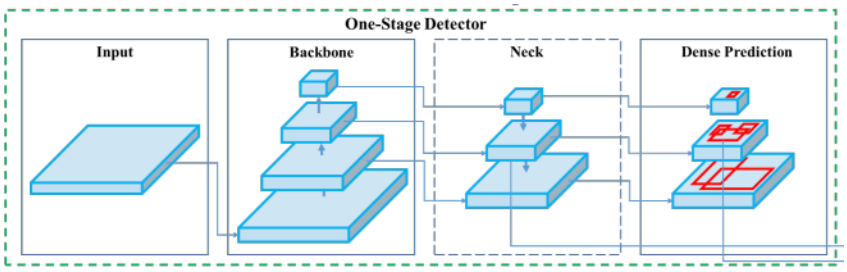


Figure 3: Basic Framework for YOLOv4 and YOLOv5

3. Proposed Method

The proposed architecture for emergency vehicle detection is based on the concept of transfer learning. Recently the standard for objection detection which include images classification and localization uses a well-trained deep convolutional neural network which are very efficient for such a task. Some of the major drawback with the deep neural network is that they require huge amount training data and corresponding labels for supervised-learning training and also enormous resources are required in terms memory and processing power of the hardware (Ashir et al., 2021). These drawbacks lead to popularization of a transfer learning whereby a pre-trained convolutional neural network with learned weights is used with little modification and fine-tuning. Our proposed approach uses a pretrained object detection deep convolutional neural network called YOLOv5 which was trained in COCO image dataset (Lin et al., 2014) with 80 output classes. The pretrained YOLOv5 is modified and partially retrained. The top layers (fully connected layers) of the YOLOv5 algorithm were re-designed with 4 output classes and retrained to get new learned weights while freezing the bottom layers (convolutional layers) and maintain the pre-trained YOLOv5 weights.

The general layout of the proposed emergency vehicle detection is shown in Figure 4. In the training phase, a database of emergency vehicles was collected and grouped into 4 categories. Then the annotation was made on the dataset using a special annotation application to identify the locations of the emergency vehicles within the image frame. The annotation information is usually extracted to text file known as sample description file which can be used during supervised learning by the deep convolutional networks to extract the positive samples from a given training image. A customized pretrained YOLOv5 is then used to partially retrain with the new dataset which four output classes (ambulances, firetruck, police, regular cars). The training parameters for the customized YOLOv5 are adjusted until a satisfactory trained model is obtained. We then develop a Graphical Interface with

FLASK libraries to integrate the trained model, camera streaming and the processing hardware to provide a real-time interface for detection of the emergency vehicle.

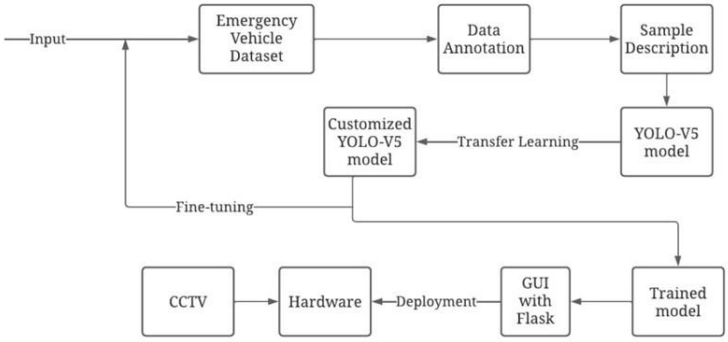


Figure 4: Flowchart of the proposed approach

3.1 Emergency Vehicle Dataset

The dataset used here is publicly available on Kaggle (SINGH, 2020). The original dataset contains a total of 2352 images which was divided into two binary classes emergency and non-emergency vehicles and was originally intended for image classification. Each image has a size of 224x224x3. To accomplish our objective, we split the emergency class into 3 classes (Ambulance, Fire truck, Police car) to make the total classes to 4 including the non-emergency vehicle as seen in Figure 5.

Figure 5: Re-grouped emergency vehicle dataset

3.2 Data Annotation

To train supervised learning algorithm such as YOLO for object detection it requires positive sample description files which will contain the spatial coordinates (object centers, width and height) and the class of the object of interest within the image. For YOLO detector the spatial coordinates or the bounding boxes around the object are normalized with respect to the entire image. A special graphical object annotation application was used to create the bounding boxes round the object and extract both the spatial coordinates and class of the object in a process referred to object annotation as shown in Figure 6.

Figure 6: Object annotation

3.3 Transfer Learning

The aim of transfer learning is to drastically reduce the time, resources and effort required to collect and label dataset, train a deep learning and processing power. Instead of creating a new deep learning architecture and collecting and labelling a huge dataset, a pretrained model which is proven in terms of its efficiency can be used with little modification to accomplish a downstream task. In this work, a pretrained YOLOv5 on COCO dataset was used with modification. We replaced the top-layer (output layer) of the network which contained 84 classes with an output layer containing 4 classes as shown in Figure 7. Since the downstream task is similar to the dataset which was used for training YOLOv5, the bottom convolutional layers of the model were frozen during the training and hence were not changed. We retrain the network with very low learning rate with the new dataset and the proposed fine-tuning to get the final version.

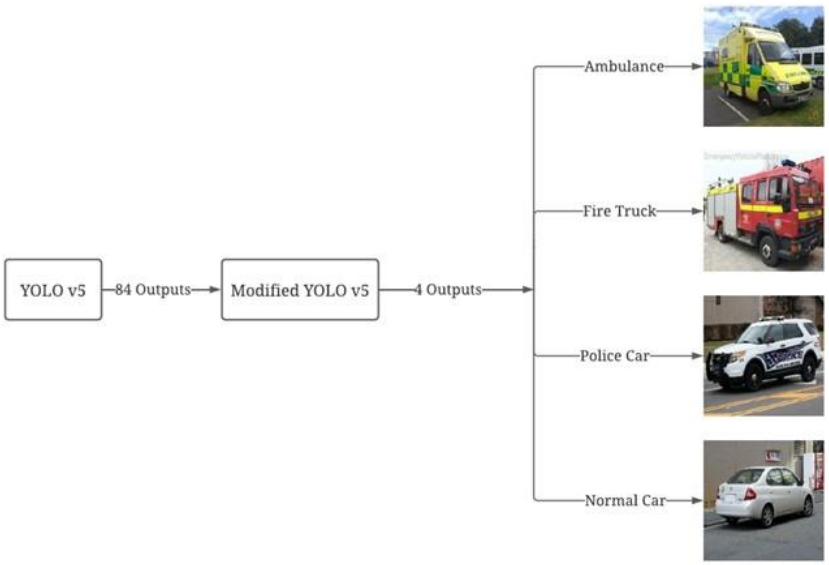


Figure 7: Modified output layer of the YOLOv5

4. Experimental Result and Discussion

In this chapter we present the results from the experiment based on the proposed approach. The dataset described in section 3.1 was used to train the modified YOLOv5 model. The dataset was split into three: training (70%), validation (10%) and testing (20%). For each sample (image) in training and validation dataset has an associated label in data description file which contains the class type of the object and its location within the image. A total of 1000 images were used with four classes for firetruck, police, ambulance, and normal cars.

4.1 Experimental Setup

Unlike most of previous version of YOLO detectors, YOLOv5 was implemented using PyTorch libraries in python. We implemented the proposed method in python environment with additional libraries such as flask, NumPy. In a windows 11 Operating System environment. Some of the hardware resources used in the experiment are listed in Table 1 below.

Table 1: Hardware resources

Desktop Computer	Dell XPS 8930 Tower
CPU	8th Generation Intel Core i7-8700k, 6 cores processor @3.7 GHz
RAM	16 GB, 2666 MHz
Storage	SSD KXG50ZNV 512G NVMe TOSHIBA DDR4 515GB
Graphics card	NVIDIA GeForce GTX 1080 8 GB
Operating System	Windows 10 Home 64-bit

4.2 Training and Validation Results

With the setup described above and approximately 8000 training and validation images, it takes roughly 4 days to complete the training with 500 epochs. During the training some important metrics were recorded. These matrices include training and validation losses; Precisions, Recall and mean Average Precision (mAP). These parameters are indicative of how well the algorithm learns the task at hand (Anwar & Ashir, 2020; Ashir, 2022). All the losses initially started very high as expected and gradually approaches to zero as the number of epochs increases. While for Precision, Recall and mAP they get better as number of epochs increases approaching their best values of 1 as can be seen in Figure 8. The box-loss represents how well the algorithm can locate the center of an object and how well the predicted bounding box covers an object. Classification loss gives an idea of how well the algorithm can predict the correct class of given object. The mean average precision (mAP) is a popular matric in measuring the accuracy of object detectors. Average precision computes the average precision value for all the instances of object within an image and over the entire collection of images. Precision measures how accurate is the predictions whereas Recall measures how good you find all the positives. Figure 8 presents the o training performance metrices over 500 epochs.

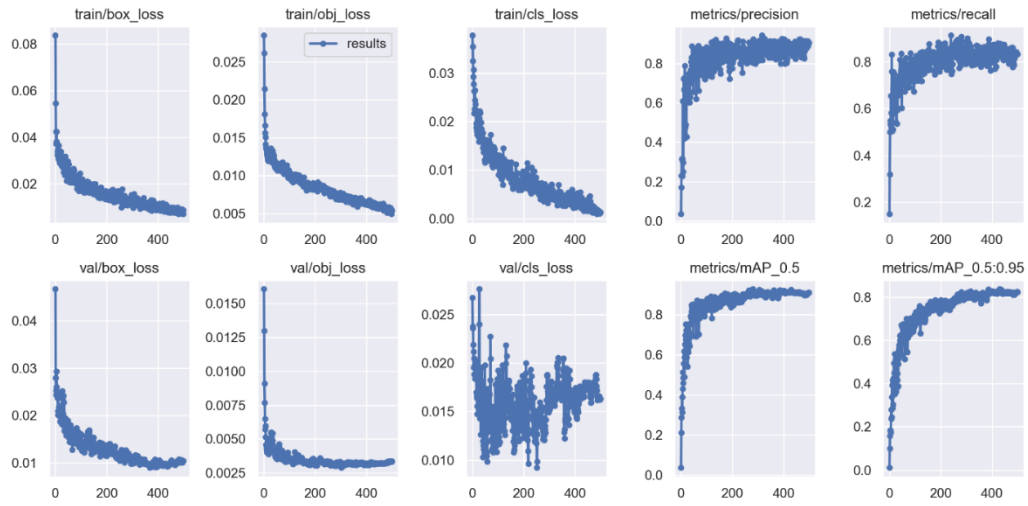
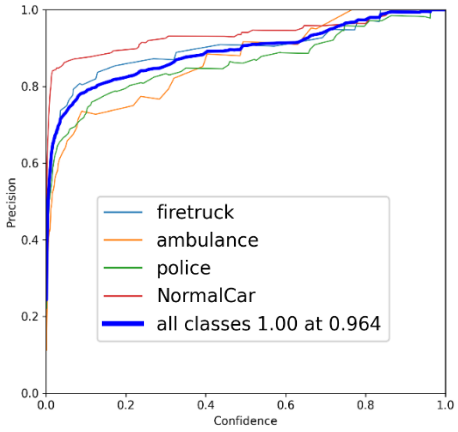


Figure 8: Training results performance evaluation

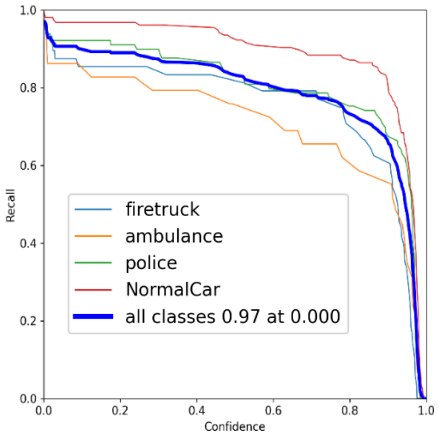
4.3 Evaluation Metrics

4.3.1 Precision, Recall and F1-score

To further examine the performances of the trained model on the individual class performance levels, the metrics which are computed during training were further analyzed and compared among themselves as presented in Figure 9-10. Figure 9(a)-(b) present the plot of the Precision and Recall against the confidence values for the individuals' classes and overall. In a similar way Figure 10 (a) depicted the F1-score (geometric mean between precision and recall) against the confidence score whereas in Figure 10 (b), we presented the comparison between the precision and recall.



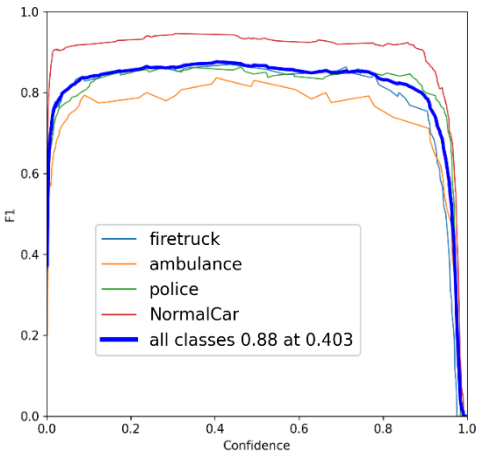
(a)



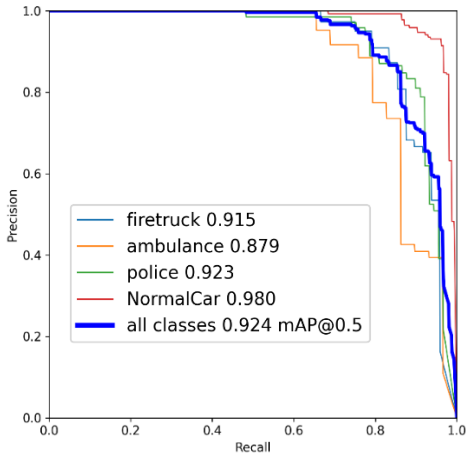
(b)

Figure 9: (a) Comparison of precision with confidence

(b) Comparison of recall with confidence



(a)



(b)

Figure 10: (a) Comparison of F1-score with confidence

(b) Comparison of precision with recall

4.3.2 Confusion Matrix

The output of our confusion matrix as shown in Figure 11 shows the 4 classes of data to be detected. The confusion matrix presents the correct classification (diagonal entries) of each class and misclassification (off-diagonal entries). More apart from the four classes, a background False Negative (background False FN) column was included which represent the percentage of false detection in each class.

Firetruck	0.81	0.03	0.01	0	0.19
Ambulance	0.04	0.76	0.04	0.01	0.03
Police cars	0.04	0.07	0.85	0.06	0.28
Normal cars	0	0.07	0.03	0.91	0.5
Background FN	0.1	0.07	0.06	0.03	0
	Firetruck	Ambulance	Police Cars	Normal cars	Background FN

Figure 11: Confusion matrix

4.4 Testing and Graphical Interface

Apart from the regular testing which follows model training, a graphical user interface was implemented using a Flask library in python. The GUI provides three key options for testing new data. These options include collection of images from a folder location, a video file or a stream of video from a live camera feed for real-time detection. Figure 12 to 13 show these options.

Figure 12: Emergency vehicle detection with the proposed algorithm

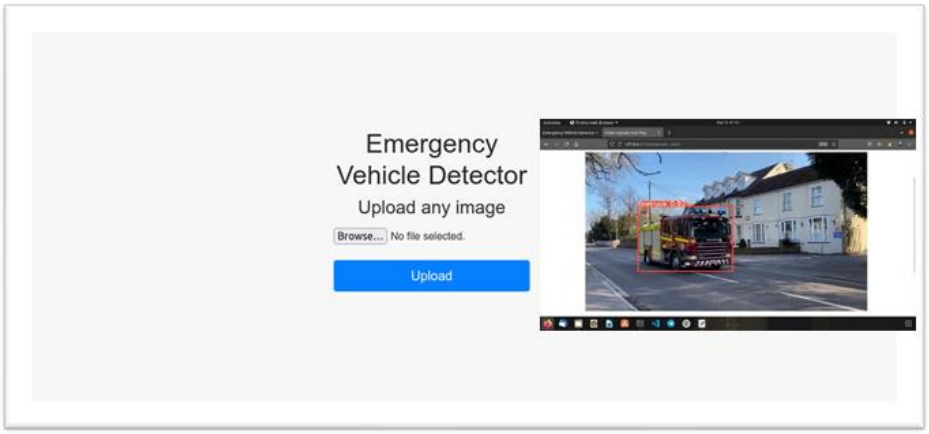


Figure 13: GUI for emergency vehicle detection with the proposed algorithm from a picture

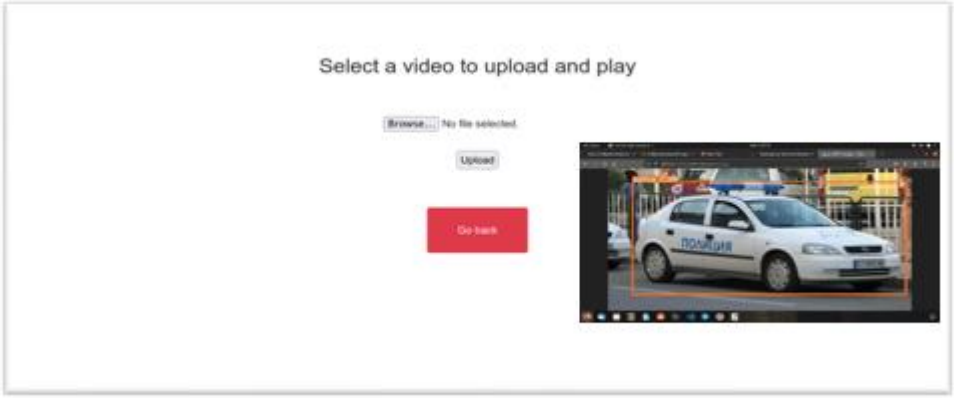


Figure 14: GUI for emergency vehicle detection with the proposed algorithm from video or camera

5. Conclusion

As the global population grows astronomically, our cities become denser and more crowded than ever.

The need for safe and secure habitat cannot be overemphasized. As our cities become smarter and transition into an era of artificial intelligence, proper and preferential handling of emergency situations within a crowded city will go a long way in easing off fatality occurrences. In this article, we proposed and implemented a real-time emergency vehicle detection system using a deep convolutional neural network. The simulations results have clearly shown that the proposed approach is practicable and efficient in detecting emergence vehicles approaching an intersection irrespective of the type, model or shape of the vehicle. If integrated into a traffic controller system, it has the capability of improving the safety, security and saving lives in emergency situations.

References

Anwar, M., & Ashir, A. M. (2020). An efficient image de-blurring technique using point spread function in high definition medical image. *Eurasian Journal of Science & Engineering*, *6*(1), 27–38.

Ashir, A.M., Ibrahim, S., Abdulghani, M., Ibrahim, A. A., & Anwar, M. S. (2021). Diabetic retinopathy detection using local extrema quantized haralick features with long short-term memory network. *International Journal of Biomedical Imaging*, *2021*. https://doi.org/10.1155/2021/6618666

Ashir, Abubakar M. (2022). Multilevel thresholding for image segmentation using mean gradient. *Journal of Electrical and Computer Engineering*, *2022*. https://doi.org/https://doi.org/10.1155/2022/1254852

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). *YOLOv4: Optimal speed and accuracy of object detection*. arXiv. https://doi.org/10.48550/ARXIV.2004.10934

Djahel, S., Smith, N., Wang, S., & Murphy, J. (2015). Reducing emergency services response time in smart cities: An advanced adaptive and fuzzy approach. *2015 IEEE First International Smart Cities Conference (ISC2)*, 1–8. https://doi.org/10.1109/ISC2.2015.7366151

Girshick, R. (2015). *Fast R-CNN*. arXiv. https://doi.org/10.48550/ARXIV.1504.08083

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2013). *Rich feature hierarchies for accurate object detection and semantic segmentation*. arXiv. https://doi.org/10.48550/ARXIV.1311.2524

Girshick, R., Iandola, F., Darrell, T., & Malik, J. (2014). *Deformable part models are convolutional neural networks*. arXiv. https://doi.org/10.48550/ARXIV.1409.5403

Glenn Jocher. (2021). *GitHub.YOLOV5-Master* (No. V5). GitHub.YOLOV5-Master. https://doi.org/https://github.com/ultralytics/yolov5/

Huang, Y.-S., Shiue, J.-Y., & Luo, J. (2015). A traffic signal control policy for emergency vehicles preemption using timed petri nets. *IFAC-PapersOnLine*, *48*(3), 2183–2188. https://doi.org/https://doi.org/10.1016/j.ifacol.2015.06.412

Humayun, M., Almufareh, M. F., & Jhanjhi, N. Z. (2022). Autonomous traffic system for emergency vehicles. *electronics*, *11*(4). https://doi.org/10.3390/electronics11040510

Karpis, O. (2012). System for vehicles Classification and emergency vehicles detection. *IFAC Proceedings Volumes*, *45*(7), 186–190. https://doi.org/https://doi.org/10.3182/20120523-3-CZ-3015.00037

Li, Z., Tian, X., Liu, X., Liu, Y., & Shi, X. (2022). A two-stage industrial defect detection framework based on improved-YOLOv5 and optimized-inception-resnetV2 models. *Applied Sciences*, *12*(2). https://doi.org/10.3390/app12020834

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common Objects in Context. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), *Computer Vision -- ECCV 2014* (pp. 740–755). Springer International Publishing.

Nellore, K., & Hancke, G. P. (2016). Traffic management for emergency vehicle priority based on visual sensing. *Sensors*, *16*(11). https://doi.org/10.3390/s16111892

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). *You only look once: Unified, real-time object detection*. arXiv. https://doi.org/10.48550/ARXIV.1506.02640

Redmon, J., & Farhadi, A. (2016). *YOLO9000: Better, faster, stronger*. arXiv. https://doi.org/10.48550/ARXIV.1612.08242

Redmon, J., & Farhadi, A. (2018). *YOLOv3: An incremental improvement*. arXiv. https://doi.org/10.48550/ARXIV.1804.02767

Ren, S., He, K., Girshick, R., & Sun, J. (2015). *Faster R-CNN: Towards real-time object detection with region proposal networks*. arXiv. https://doi.org/10.48550/ARXIV.1506.01497

Roy, S., & Rahman, M. S. (2019). Emergency vehicle detection on heavy traffic road from CCTV footage using deep convolutional neural network. *2019 International Conference on Electrical, Computer and Communication Engineering (ECCE)*, 1–6. https://doi.org/10.1109/ECACE.2019.8679295

Simonyan, K., & Zisserman, A. (2014). *Very deep convolutional networks for large-scale image recognition*. arXiv. https://doi.org/10.48550/ARXIV.1409.1556

Singh, A. (2020). *Emergency vehicles identification dataset*. Kaggle. https://www.kaggle.com/datasets/abhisheksinghblr/emergency-vehicles-identification

Sun, H., Liu, X., Xu, K., Miao, J., & Luo, Q. (2021). *Emergency vehicles audio detection and localization in autonomous driving*. arXiv. https://doi.org/10.48550/ARXIV.2109.14797

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). *Rethinking the inception architecture for computer vision*. arXiv. https://doi.org/10.48550/ARXIV.1512.00567

Tran, V.-T., & Tsai, W.-H. (2020). Acoustic-based emergency vehicle detection using convolutional neural networks. *IEEE Access*, *8*, 75702–75713. https://doi.org/10.1109/ACCESS.2020.2988986