Enhanced Driver Assistance: Optimizing Computer Vision for Emergency Vehicle Detection

Tony Zheng
MDSAI, University of Waterloo
200 University Ave W, Waterloo, ON N2L 3G1
t28zheng@uwaterloo.ca

Neeraj Nagar
MASc, Electrical and Computer Eng.
200 University Ave W, Waterloo, ON N2L 3G1
nnagar@uwaterloo.ca

Figure 1: Jeremy Clarkson's Porsche 944 Ambulance, Top Gear Season 22, Episode 3.

Abstract

To ensure safety on public roads, yielding to emergency vehicles is a top priority for drivers when they see flashing emergency lights and hear sirens. However, for many reasons, emergency vehicles may forgo siren use during an emergency, instead opting to use only their flashing lights. In some cases, this may not be enough to alert a human driver in time such that the driver is not in the way of the emergency vehicle. The primary goal of this project is to improve the real-time detection and classification of emergency vehicles (e.g., ambulances, fire trucks, police cars, and snow plows) and determine their operational status (responding or not) using computer vision and machine learning techniques. By leveraging existing models and fine-tuning them for this specific task, we aim to design models suitable for real-time applications, particularly in driver assistance systems that may improve driver readiness and responsiveness. In this study, we used the efficiency of the YOLOv8 Nano model for the object detection task due to the extremely competitive speeds offered by the YOLO architecture. Compared to its predecessors YOLOv1 through YOLOv7, YOLOv8 achieves superior accuracy while also maintaining a smaller model size[7], allowing it to be deployed in resource-constrained systems such as those in nonautonomous vehicles. Furthermore, when YOLOv8 is compared to YOLOv11, the tests reveal that YOLOv8 is faster and offers comparable detection performance in this application. This project aims to contribute to safer roads and more responsive driver assistance technologies.

1 Introduction

In high-stress driving environments, being able to respond quickly to surrounding emergencies is the key to ensuring road safety. However, drivers often fail to notice emergency vehicles, such as police cars or ambulances approaching from behind. This is especially the case in conditions where there is no audible siren present. This situation can lead to delayed responses and dangerous situations. While modern driver assistance systems can help address this problem, ~~but~~ they are often resource-constrained systems that make the design of a lightweight model more important.

This study focuses on the development of a robust model capable of identifying emergency vehicles and snow plows, both vehicles that must be yielded to, and determine their operational status through whether or not their lights are flashing. The primary objective is to create a model that can be applied to current vehicle systems, providing real-time assistance without excessive computational overhead that is less consequential in driverless vehicles with more powerful processors. Unlike previous efforts which used more traditional non-AI methods to detect sirens [1], this project uses a neural network for detection and incorporates a broader range of vehicles and augmentations to simulate a broader range of weather conditions.

To achieve this goal, we used the YOLOv8 Nano architecture, chosen after testing various YOLO architectures due to its speed and accuracy balance. Similar previous work in this domain involved YOLOv7 [1], however, YOLOv8 had already made its advent between the conclusion of that work and the beginning of this one. In our research, we noted that appropriate datasets for this application were not fully established, which is not an unlikely issue when taking novel approaches to machine learning. Therefore, a significant portion of the effort was dedicated to data preparation, including manually labeling over 2600 images to ensure the quality and readinessof the dataset.

This study is structured as follows: first, we present the background for this project, highlighting the challenges and goals. Next, we discuss the selection of the YOLOv8 model, including comparisons with alternative architectures. We then detail the data collection and augmentation process, emphasizing the constraints and solutions employed. This is followed by a performance evaluation where metrics such as inference speed and mAP are analyzed. Finally, we discuss the results and unforeseen advantages of our system, and detail potential areas for future achievement.

This project aims to address the need for a lightweight, real-time detection system that covers a wider variety of vehicles while also being able to run utilizing constrained resources. It must be noted that while autonomous vehicles can simultaneously process audio and visual data [6], we will work solely with visual input.

2 Background and Related Work

Object detection is a computer vision process that involves identifying objects within an image or video, and then identifying the location of the object using bounding boxes. This process combines classification (determining what an object is) with localization (determining where the object is). This process is often used in applications such as autonomous driving and robotics.

While initial models for object detection have been established in the past, they often suffer from performance limitations, including slow processing speeds, high latency, and substantial computational resource requirements. As a result, we explored the most commonly used and well-supported architectures, finally having to choose between Faster R-CNN and YOLO (You Only Look Once). Previous research provided evidence suggesting the YOLO model had faster inference while having comparable accuracy to Faster R-CNN [3], finalizing our decision to implement our system using the YOLO model.

YOLO is fundamentally different from models like Faster R-CNN in how it approaches object detection. While Faster R-CNN uses a two stage process that involves generating "region proposals" and then classifying and refining those regions, YOLO uses a single stage. YOLO divides the input into a grid and directly predicts bounding boxes and certainty scores for each cell. This design makes YOLO significantly faster and better for real-time object detection, at the cost of accuracy (the opposite of Faster R-CNN's approach).

The most similar approach to our study is an experiment conducted by Github user Dynle [1], where transfer learning is used to fine-tune YOLOv7 models to adapt to the process of image recognition. However, Dynle's project differs in the techniques used to identify whether or not an emergency vehicle is active, using an algorithmic solution to detect changes in contrast in the vehicle's emergency lighting. In contrast, we use a YOLO model to conduct this detection. Additionally, the models trained by Dynle are tuned to be very effective at identifying emergency vehicles in Asia, while the goal of our study is to develop a model effective in North America.

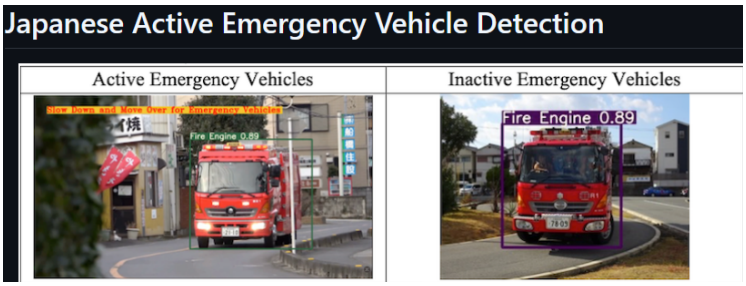

Figure 2: An example of inference performed by Github user Dynle's YOLOv7 model. [1]

In summary, while previous efforts have achieved notable progress in vehicle detection, their limitations in scope and adaptability leave room for improvement. This project aims to address these gaps and show that significant improvements can be made to the task of emergency vehicle detection.

3 Data

3.1 Data Collection

The dataset used to train these models is a combination of multiple datasets from multiple sources. Due to one of the approaches requiring 2 different models with 2 different object detection tasks, 2 datasets were curated - one for special vehicles themselves, and the other for siren identification.

In the dataset for special vehicles, there were on average 1500 samples of each class in the training set: "emergency" for an emergency vehicle, "non-emergency" for a non-emergency vehicle and "snowplow" for snow plows, etc. The images of non-emergency vehicles and emergency vehicles were sourced from an anonymous user's Roboflow dataset [4], while the snowplow images were taken from images.cv [2].

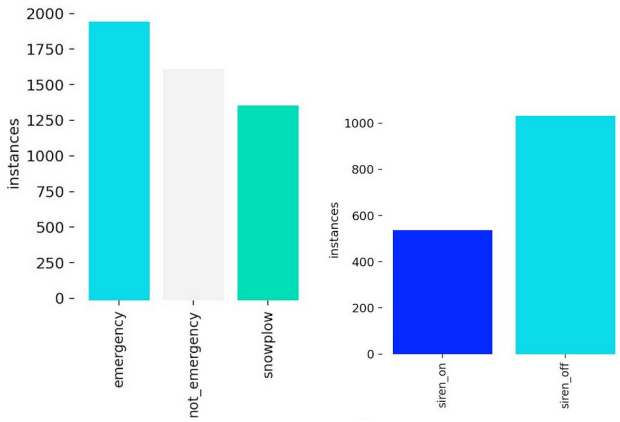

Figure 3: The distribution of the classes within the datasets used to train the models.

In the dataset for siren light recognition, there were about 500 images of sirens flashing and 1000 images of sirens not flashing[5]. It's possible that this imbalance of the dataset could lead to decreased

accuracy, but we opted to use all of the available data given its scarcity. The less represented data was not augmented in the interest of not overfitting the images in the underrepresented class.

3.2 Data Annotation

It must be noted that a significant portion of the data preparation effort was dedicated to data annotation, with labels manually annotated using the software LabelImg. Around 2600 of the images across both datasets had either corrupted or incorrect labels, resulting in the necessity of manual correction.

Figure 4: Examples of class annotations and their bounding boxes. The box is drawn around the relevant subject, and given a numerical label that corresponds to some text label.

3.3 Data Augmentation

Several augmentations were applied to a random subset of images in order to improve the robustness of our models. The pre-augmented images were included in the dataset with the augmented ones. Before augmenting the data, the data was resized to 416 x 416 through stretching. This matches the input dimensions of our model. Table 1 describes the augmentation performed on the datasets before they were used for training:

Table 1: Description of Applied Data Augmentations

Augmentation	**Description**
Random Rotation	Rotates the image randomly between -15 degrees and +15 degrees, simulating variations in camera angles.
Random Shear	Skews the image horizontally and vertically by up to 15 degrees, mimicking perspective distortions.
Random Exposure	Adjusts brightness from -25% to +25%, simulating diverse lighting conditions.
Random Gaussian Blur	Applies a blur effect with intensity between 0 and 1.25 pixels, imitating motion or focus inconsistencies.
Salt and Pepper Noise	Adds noise to 5% of pixels, turning some randomly black ("pepper") or white ("salt"), to simulate low-quality images.

The main goal of these augmentations was to construct a dataset that would help increase the robustness of the model, and to increase the number of training samples.

3.4 Challenges and Limitations

A key limitation of this dataset was that all subjects were in the foreground and close to the frame, creating an overrepresentation of vehicles being present front and center. Unfortunately, this shortcoming could not be addressed without acquiring further data, which could only be reasonably done given more time for proper data research and exploration.

4 Model Architecture

In this study, the architecture of the model remains unchanged from the released version of YOLOv8 [7]. We will describe the architecture briefly.

The YOLOv8 nano (or YOLOv8n) variant of YOLOv8 is an optimized, lightweight version of the architecture, designed specifically for scenarios with limited computational resources. It is composed of two parts: a backbone network and a detection head.

The backbone network of YOLOv8 is based on EfficientNet-B4, which is a scaled up form of the original EfficientNet. This is a convolutional neural network with 29 layers and includes modified residual blocks with squeeze and excitation modules. This backbone network outputs five feature maps with diverse resolutions and dimensions, and passes these feature maps to the detection head for further processing.

The detection head of YOLOv8 is a NAS-FPN-Cell architecture, which is a sub network with six layers and 256 channels. After NAS-FPN-Cell takes the five feature maps generated by the backbone network, it applies a series of fusion operations and connections to them. These operations include element-wise expansion, element-wise multiplication, global average pooling, max pooling, and concatenation. The connections link the diverse features across the feature maps. Then, NAS-FPN-Cell outputs five feature maps that are then utilized to generate bounding box predictions.

4.1 Improvements over YOLOv7

YOLOv8 has several improvements to the previous YOLO variants, such as a new loss function, a different augmentation method and a different evaluation metric. These improvements are meant to improve the execution and robustness of the algorithm, and addresses the limitations of previous variants.[7]

4.2 How YOLOv8 was used

In this work we experimented with two setups, both involving two YOLOv8 models. The first setup was a sequential model, where two models were placed one after another. The 416 x 416 image was passed as an input to the first model (the vehicle identifying model), which would identify any vehicles of interest and provide its bounding boxes. Next, the process would crop out this region of interest and pass it to a second model, tasked with identifying active sirens.

The second setup also uses two models, but instead of the vehicle-identifying model pre-cropping the input to be passed to the siren-identifying model, it is simply used to confirm the existence of a vehicle in the input image, reducing the siren model's risk of a false positive.

Brief experimentation was also conducted with the YOLOv11 model, but the inference speed was simply too slow compared to YOLOv8 (about ten times slower).

5 Results

5.1 Evaluation Metrics

In object detection, there are several unique evaluation metrics to be aware of. We will briefly explain them below.

5.2 Lightbar Model

The model for detecting the active siren of a vehicle, named the "Lightbar Model", was trained on a dataset entirely labelled by hand. Although this means that it is highly unlikely for any samples to be corrupted or to be labeled inappropriately, the low number of total samples likely was a factor that reduced the models' ability to generalize on the unseen data in the test set.

Table 2: Evaluation terms

Term	Description
Box loss	Refers to the error associated with predicting the bounding box coordinates for an object in an image. Considers both the overlap of the predicted and ground truth boxes as well as the distance between their centers.
Class loss	Measures how accurately the model predicts the class labels of the detected objects. Calculated by comparing the predicted class probabilities against the true class (typically with a softmax operation).
Distillation loss	A form of knowledge distillation applied during training to reduce the discrepancy between teacher-student models. This loss helps distill bounding boxes over multiple box predictions to fine tune the model's box prediction.
mAP50	Mean Average Precision at 50% Intersection over Union. This means the bounding box is counted as correct if the percentage of intersection is at least half the percentage of the total pixels covered by the union of the ground truth and the prediction box.
mAP50-95	Extends mAP50 by averaging precision scores over multiple thresholds. The thresholds range from 0.5 to 0.95, providing a better insight into the model's performance over the mAP50 statistic.

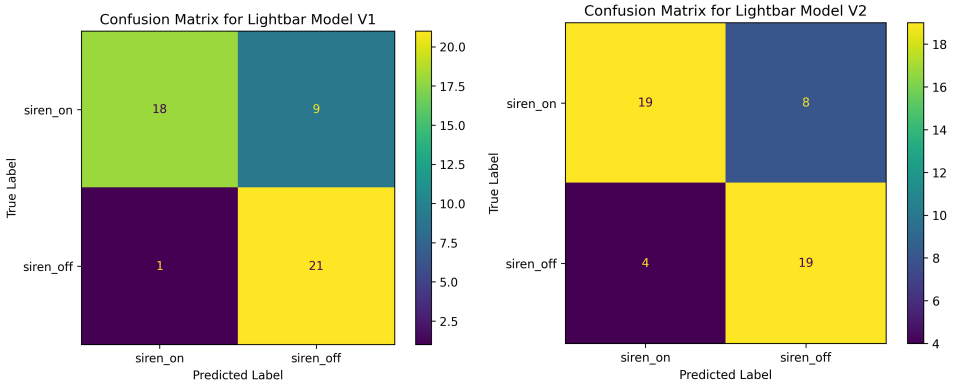

Figure 5: Confusion Matrices of Lightbar Model V1 and Lightbar Model V2 on the same test set, respectively.

Two models were trained on the same training set. Lightbar Model V1 was trained for 100 epochs, while Lightbar Model V2 was trained for 250 epochs, but did not improve in validation loss after epoch 200. The models were manually evaluated on the test set, due to the nature of automatic evaluation methods not fitting the application. The manual evaluation process involved marking any detection of a siren on when a siren in the image was on, a true positive. Conversely, if a siren was off and it was either detected as off or not detected at all, a true negative. Evaluating in this manner allows us to determine which model fits closest to the application of detecting a running siren.

Table 3: Comparison of Metrics Between Lightbar Detecting Models, final Validation accuracy

Metric	Model V1	Model V2
mAP50	0.78211	0.85746
mAP50-95	0.46928	0.50166
Box Loss	1.77020	1.97903
Cls Loss	1.29507	1.84269
DFL Loss	1.58477	1.76388

Due to technical difficulty, final metrics were calculated on a unique validation set instead of a dedicated test set. On initial inspection, it appears that while Model V2 has better performance drawing bounding boxes, it appears to have significantly worse class loss than the lesser-trained

Model V1. Indeed, in the test set, Model V1 achieves a slightly higher class accuracy (78%) than Model V2 (76%). Therefore, we will plan to use Model V1 in our ensemble model.

5.3 Vehicle Model

The model for identifying emergency vehicles and snowplows, ~~or the~~ Vehicle Model, is the one that will work either sequentially or in parallel to the Lightbar Model. This model was trained for 150 epochs, and the weights were acquired from the iteration with the best validation loss.

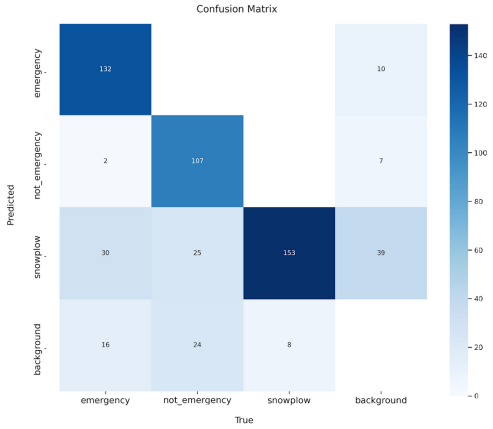

Figure 6: Confusion Matrix of the Vehicle detection model on the test set.

This model was not manually evaluated, instead relying on automatic evaluation that may have been less lenient with contextual clues. For example, in images where both a normal vehicle and a snowplow were detected, it is unclear which class the automatic evaluation would assign the model's answer. Therefore, the results were not entirely clear for the accuracy of the vehicle model. However, it is evident that the vehicle model is more willing to identify the class as a snowplow more than any other. This is evident in the high number of false positives in the snowplow class over all other classes.

Table 4: Final Validation metrics of Vehicle Model

Model	mAP50	mAP50-95	Box Loss	Cls Loss	DFL Loss
Vehicle Model	0.63686	0.48226	0.97786	1.0329	1.26526

6 Discussion

6.1 Interpreting the Results

The trained models did not perform as expected. Even for the best lightbar model, there were many false negatives where a visual, running siren would not be detected whereas a human would reliably detect the siren. It is likely that the limited dataset of 1500 samples affected the model's ability to generalize, but an accuracy of 78% does not meet our desired threshold for reliable emergency detection in real time.

In regards to the vehicle model, the results were also suboptimal. The model seemed very likely to predict the snow plow class, despite that class being in the minority compared to the others. It may be the case that the model learned to predict the elements surrounding the snowplow, such as the large amount of snow, instead of the snowplow itself. It must also be noted that during manual labeling, it was often the case that the snowplow class would not be completely within the frame, likely leading to the model learning an ambiguous definition for the snow plow and being biased to larger frame sizes.

Figure 7: An example of a poorly framed Snow plow image. Most of the vehicle is not in sight.

While previous work such as that done by Dynle produced much more promising results [1], their model was trained and evaluated on video files as their primary data source, allowing the models to leverage temporal information across consecutive frames. This approach provides several advantages, particularly in tasks like emergency vehicle detection where temporal patterns such as flashing lights or movement can enhance accuracy (in contrast to the approach taken in this study).

While the image-based approach is lightweight, it may be more effective to consider temporal data in future iterations of the experiment.

6.2 Interesting Advantage

An unforeseen advantage of the approach taken in this study is the model's ability to detect unmarked emergency vehicles as soon as their lights are activated. Unlike methods reliant on vehicle appearance, this model focuses on the presence of active sirens or lightbars, making it effective even for police cars without outward identifiers. This capability is particularly valuable in real-world scenarios and allows for a timely detection and response irrespective of the vehicle's initial appearance.

Figure 8: An image of an unmarked Kia Ceed, a vehicle typically not used for law enforcement. The lightbar model correctly identifies the Kia's running siren lights.

This ability reveals a promising avenue for future work to pursue unmarked vehicle detection before the vehicles reveal themselves through their emergency lighting.

7 Conclusion

In conclusion, in this study we explored the development and evaluation of an ensemble model to detect emergency vehicles and snowplows, with the goal being to identify whether they are active or not through running siren lights. While the model demonstrated moderate success, its overall performance was limited by the size and quality of the dataset, and the lack of temporal information. Despite these challenges, the study still demonstrates the potential for real-time detection with constrained resources, and has the unforeseen advantage of detecting unmarked vehicles when they activate their emergency lights, offering another way to enhance driver awareness and safety on the road.

References

[1] Dynle. Dynle/japanese-emergency-vehicles-detection: A transfer learning based algorithm using yolov7 for active emergency vehicle detection and classification in japan. https://github.com/dynle/japanese-emergency-vehicles-detection/tree/main.

[2] images.cv. Snowplow image dataset. https://images.cv/dataset/snowplow-image-classification-dataset.

[3] Fiza Joiya. Object detection: Yolo vs faster r-cnn. *International Research Journal of Modernization in Engineering Technology and Science*, Sep 2022.

[4] project sawkw. Emergency vehicle detection dataset. https://universe.roboflow.com/project-sawkw/emergency-vehicle-detection-el8ej, may 2022. visited on 2024-11-27.

[5] Emergency Siren. Emergency-siren dataset. https://universe.roboflow.com/emergency-siren/emergency-siren, mar 2024. visited on 2024-12-08.

[6] The Waymo Team. Recognizing the sights and sounds of emergency vehicles. https://waymo.com/blog/2017/07/recognizing-sights-and-sounds-of/, Jul 2017.

[7] Rejin Varghese and Sambath M. Yolov8: A novel object detection algorithm with enhanced performance and robustness. In *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, pages 1–6, 2024.