CS 6476 Project Group 34

Obstacle Identification for Autonomous Vehicles

View on GitHub Download .zip Download .tar.gz

Team Members

- Sreeranj Jayadevan
- Hriday Harlalka
- Divij Mishra

The code for this project can be found in the following repository: (Git Repo)

Table of Contents

- Introduction
- Related Work
- Methods and Approach
- Experiments
- Results
- Discussion

Introduction

The goal of our project is to create a reliable system that accurately identifies the presence of and classifies different types of obstacles in the path of an autonomous vehicle. Detecting obstacles for autonomous driving is vital in advancing technology and ensuring safety on the roads with self-driving cars. In addition, classifying obstacles into more vulnerable categories, e.g. traffic cones vs pedestrians, is important to develop safer, more intelligent algorithms for autonomous vehicles. The metric we want to use and focus is on event/obstacle criticality. Also child and adult pedestrian perspectives vary widely for autonomous vehicles (Paper). Autonomous driving vehicles must take into account this safety aspect and employ suitable metric (brake reaction time, emergency braking etc.. Paper) once it classifies an obstacle. We plan to first classify various obstacles and then decide on the event criticality based on the obstacle. Since we can tolerate false positives but we don't want false negatives when it comes to safety aspects, we report **recall** as our metric. This will be studied and compared with different experiments.

We plan to develop a robust and accurate Computer Vision algorithm, that given an image, will be capable of identifying diverse types of obstacles, such as different types of pedestrians, animals, vehicles, barriers, etc.. We plan to use the concept of Transfer Learning to fine-tune a DCNN that will help solve this problem without expending an extraneous number of resources to correctly identify and classify the obstacle in an image. As part of this project, we also aim to gain experience in working with perception datasets used for autonomous driving.

Related Work

The first step is to classify different possible obstacle for an autonomous vehicle using existing frameworks. Since we are focusing on event criticality, we will be mainly focusing on classifying different types of pedestrians, pets etc.. We plan to employ existing deep learning techniques for this part. Based on the type of obstacle we are focusing on, we will be updating the different categories of obstacles we are interested in. In this regard, we will be using Transfer Learning due to its reduced training time, improved neural network performance, and the absence of a large amount of data.

Obstacle classification using Deep Learning Techniques

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25.

> This was the original AlexNet paper that kicked off the ImageNet classification trend, leading to modern DCNN architectures trained on GPUs for image recognition.

Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556 (2014)

> This is the paper that proposed VGGNet, one of the models we plan on using. They utilized large stacks of 3x3 convolutional filters to learn more complex features and improve performance.

K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

> This is the paper that proposed ResNet in 2015, one of the models we plan on using. They popularized the concept of residual connections in DCNNs, allowing them to mitigate the problem of vanishing gradients and train deeper convolutional neural networks, learning more complex image features.

Transfer learning: Thu, M., Suvonvorn, N. and Kittiphattanabawon, N., 2023. Pedestrian classification on transfer learning based deep convolutional neural network for partial occlusion handling. International Journal of Electrical and Computer Engineering (IJECE), 13(3), pp.2812-2826.

> This paper compares the various pre-trained deep learning models that were used for Transfer Learning, and highlights the impressive performance of the deep ConvNet

(Convolutional Neural Network) designs to recognize patterns of pedestrians even when they are partially hidden. The network, trained on examples containing partially covered or obscured parts of pedestrians, can extract the information and fine-tune itself. Use a dataset from Southern Thailand (PSU dataset). They append 3 linear layers at the end as the prediction head, and finetune these on their data. They use VGG16, Inception V3, ResNet50, and MobileNet as the pretrained backbones.

Shaha, M. and Pawar, M., 2018, March. Transfer learning for image classification. In 2018 second international conference on electronics, communication and aerospace technology (ICECA) (pp. 656-660). IEEE.

The above paper is about the concept of Transfer Learning in general. It highlights its impressive performance even when compared to hybrid models and other CNN models when performing image classification tasks on state-of-the-art databases.

Akhand, M.A.H., Roy, S., Siddique, N., Kamal, M.A.S. and Shimamura, T., 2021. Facial emotion recognition using transfer learning in the deep CNN. Electronics, 10(9), p.1036.

This paper above explores the usage of pre-trained DCNN for image classification task of Facial Emotion Recognition (FER). We plan to use this concept for our use case, with our training data. They fine-tune the last conv block and the linear layers. They also point out that more conv blocks might have to be fine-tuned if the task is more and more dissimilar from the original pre-training task.

Applications in Autonomous Driving

Amir Rasouli, Iuliia Kotseruba and John K. Tsotsos, York University, Toronto, Ontario, Canada. Are They Going to Cross? A Benchmark Dataset and Baseline for Pedestrian Crosswalk Behavior. International Conference on Computer Vision Workshops (ICCV Workshops)

They use a perception dataset similar to ours to predict whether a person is likely to cross a road. Their methodology also involves the use of DCNNs to identify features in the image that are significant predictors to someone crossing the road.

Methods and Approach

Overview

Our current approach is to frame the problem as a binary classification problem. The nuImages dataset comes with various annotated categories – animals, pedestrians subclasses, vehicle subclasses, obstacle subclasses. By combining animals, all pedestrian subclasses and two-wheeler vehicles into one class, vulnerable, and other classes into one class, non-vulnerable, we reframe the problem as single-label binary classification problem where each image either contains a vulnerable object or not.

Till now, our best solution to this problem has been to use a pretrained ResNet-50 model and

fine-tune the linear classifier on 10k data with early stopping. The ResNet-50 model consists of 50 convolutional layers across 5 convolutional blocks, each operating on a different image size. The convolutional blocks are finally followed by an average pool layer, and a 1000 dimensional linear layer. Instead of the 1000-d linear layer, we add a 512 dimensional linear layer, followed by a K dimensional linear layer, where K is our number of labels (K = 25 for the multi-label case, and 1 for the single-label case binary classification case).

This is what the network looks like, from the resnet paper

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Source: ResNet paper

This has given us **89%** recall and **84.5%** F1-score. We emphasize the importance of recall in this scenario as we prefer false positives over false negatives, as it is imperative that we detect any vulnerable living beings in the path of the car.

Contribution

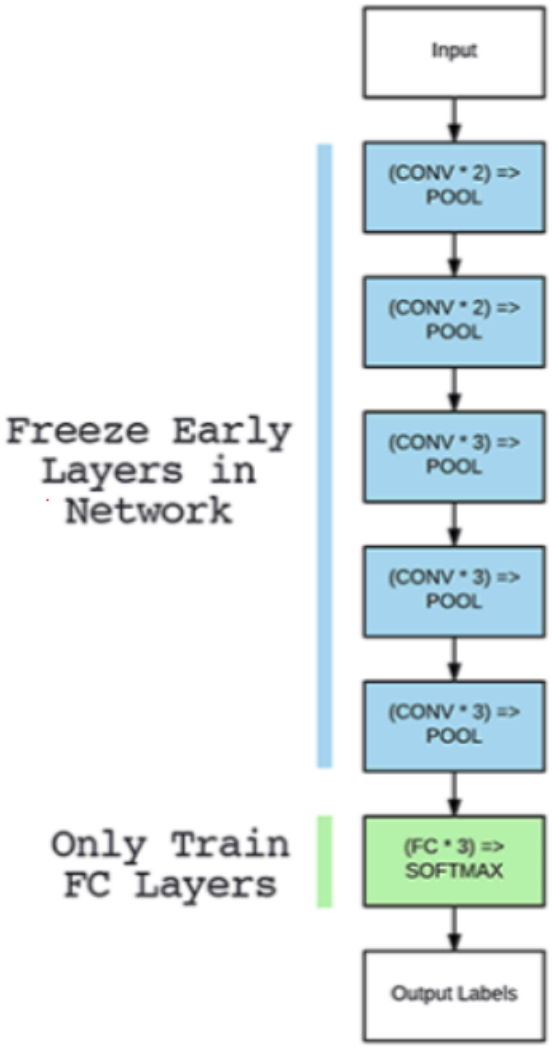
As the nuImages dataset is annotated for segmentation and object detection, we were unable to find any work focusing on image classification experiments with this dataset. That said, as image classification is an easier task, we expect to get better metrics than segmentation and object detection here. This is important, as better metrics would lead to better detection of obstacles in the vehicle's path, which could be used for more reliable decision-making.

Intuition

ImageNet-pretrained models tend to have good image feature generation ability and are well-suited for fine-tuning on new image datasets. As such, we expect our method to be successful in identifying and classifying obstacles in our dataset.

Method Visualisation

Source: pyimagesearch.com



Experiments

We had 3 major experiments.

Experiment 1

In our initial experiment, we treated the problem as a multi-label binary classification problem, i.e. we had 25 classes, and each image had 25 0/1 annotations – one for each class. We used Binary Cross-Entropy Loss for our model training, which treats each label as an independent binary classification problem. We decided to compare 3 factors:

- Do we get better performance if change the pretrained backbone from ResNet-50 to VGGNet-16?
- Are we able to get better performance if we use 10k training data vs 40k training data?

- Do we get better performance by tuning just the linear layers or also tuning some convolutional layers?
    - In the latter case, for both ResNet-50 and VGGNet-16, we decided to tune the last 1/4th convolutional layers. For ResNet-50, this meant tuning the last convolutional block, “layer_4”, while for VVNet-16, this meant tuning layers 24, 26, 28.

Experiment 2

In the previous set of experiments, most classes were present in a small subset of data, leading to low weighted recall values (~50%). To solve this, we decided to merge classes in the next set of experiments. We combined all pedestrian classes, animals, bicycles, and motorbikes, into one class, “vulnerable”. Thus, the new problem is single-label binary classification – given an image, is a vulnerable object present?

Additionally, we observed that there was no significant difference between ResNet-50 and VGGNet-16, so we stuck to using ResNet-50 for the next set of experiments. Due to time constraints, we decided to stick to 10k training data.

We added one more experiment – observing our data, we noticed that most vulnerable class objects are quite small and might be hard to detect in our downsampled 224x224 images. As such, we also ran experiments with 448x448 downsampling, hoping for recall improvement.

Experiment 3

In the above experiment, we ran multiple learning rates separately with early stopping with a patience of 5 epochs. However, we saw that validation losses tended to plateau quickly, so for our third experiment, we used a patience-based learning scheduler. Our scheduler decreases the learning rate by sqrt(0.1) if the validation loss doesn’t improve for patience = 3 consecutive epochs.

Input and Output Description

We experiment on the nuImages dataset (ref), a large-scale autonomous driving image dataset annotated with 2D boxes and segmentation masks, though we only use annotation labels for image classification. nuImages consists of 93k images split into 77k training and 16k validation images. The nuImages dataset contains high-resolution (1600x900), annotated 2D images covering many challenging driving situations relevant to our problem of interest.

Each image has annotations for 25 classes – animal, human.pedestrian (subclasses: adult, child, construction_worker, personal_mobility, police_officer, stroller, wheelchair), movable_object (subclasses: barrier, debris, pushable_pullable, trafficcone), static_object.bicycle_rack, vehicle (subclasses: bicycle, bus.bendy, bus.rigid, car, construction,

ego, emergency.ambulance, emergency.police, motorcycle, trailer, truck), and flat.driveable_surface. Later, we combined animal, all human.pedestrian classes, vehicle.bicycle and vehicle.motorcycle, into a single class “vulnerable”, clubbing the rest into “non-vulnerable”.

An example image is the following. The dataset annotations for this are [‘vehicle.car’, ‘vehicle.car’, ‘vehicle.car’, ‘human.pedestrian.adult’, ‘vehicle.truck’, ‘vehicle.truck’, ‘human.pedestrian.adult’] (the second human is near the blue sign on the right!). For our first experiment, multi-label classification, we took the annotation as [‘vehicle.car’, ‘vehicle.truck’, ‘human.pedestrian.adult’], taking each class exactly once. For later experiments, single-label binary classification, the label for this would simply be ‘vulnerable’ since there’s a human.pedestrian.adult in this image. Computationally, the former would be represented as a multihot vector (vector of num_classes 0/1s) and the latter would be represented as 0/1.

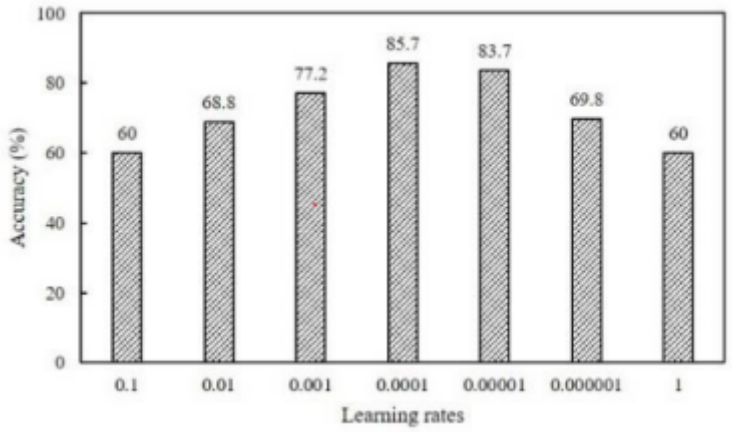
True: [‘vehicle.car’, ‘vehicle.truck’, ‘human.pedestrian.adult’]
Pred: [‘vehicle.car’, ‘vehicle.truck’, ‘human.pedestrian.adult’]

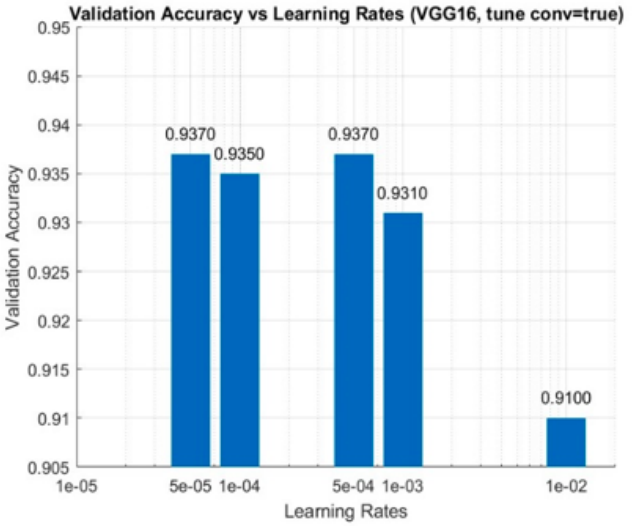
Metrics

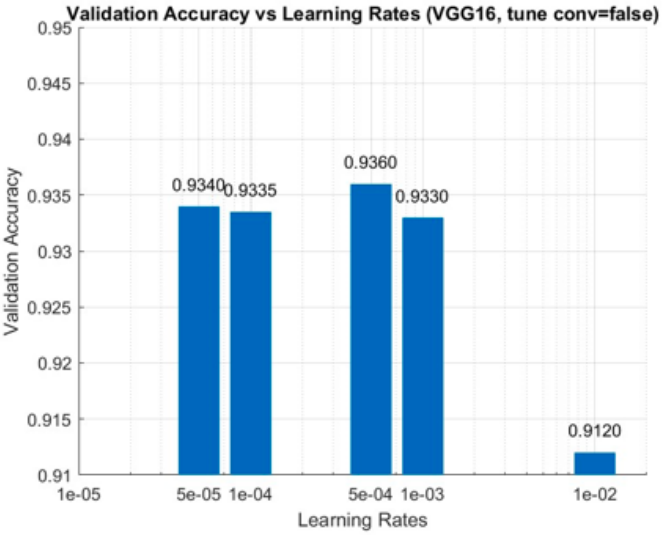
For the multi-label problem, we assigned custom weights to each class, assigning higher weights to vulnerable classes. Using these custom weights, we calculated weighted precision, recall, F1-score. For the single-label problem, we used accuracy, precision, recall, F1-score, with recall being of most importance.

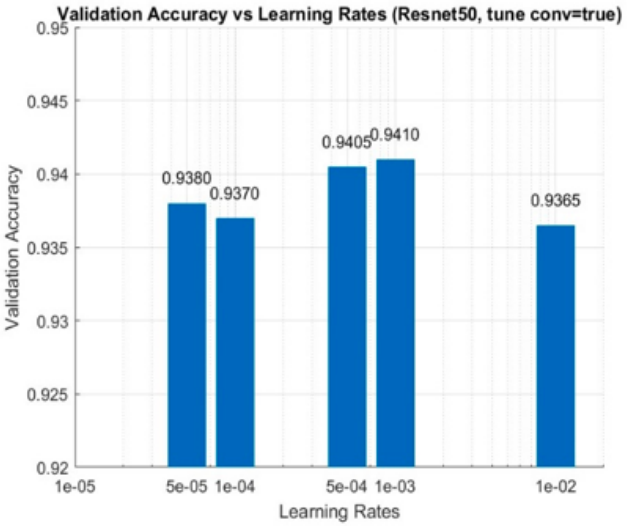
Results

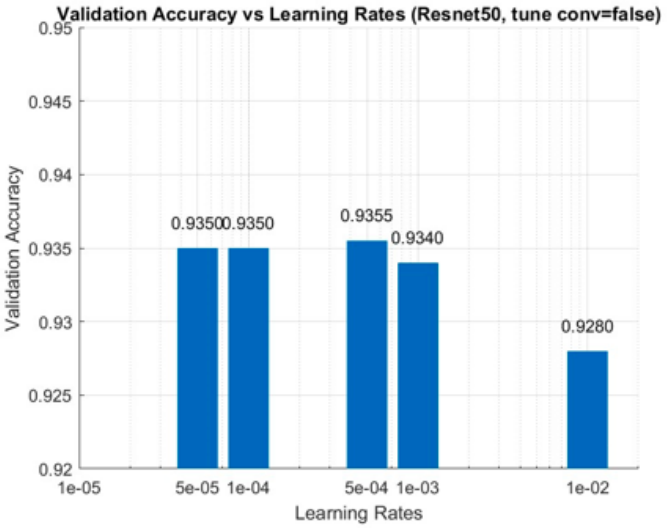
We experimented with Resnet50 and VGG16 architectures. The paper “Pedestrian classification on transfer learning based deep convolutional neural network for partial occlusion handling” did a comparative study of prediction accuracy for pedestrains using different optimisers. They also did a comparative study with different learning rates for a particular optimiser. The results showed that they got an accuracy of 85.7% for Adam optimiser with a learning rate of 0.0001. Similarly we also initially did a comparative study to understand what learning rates can give the best results. One difference with the mentioned paper is the dataset being used. The transfer learning paper used the INRIA person data set (Link), however this was not specific to applications in autonomous driving and we needed more data for our use case. We hence used the nuscenes dataset. Below are the plots from this paper and from our experiments.





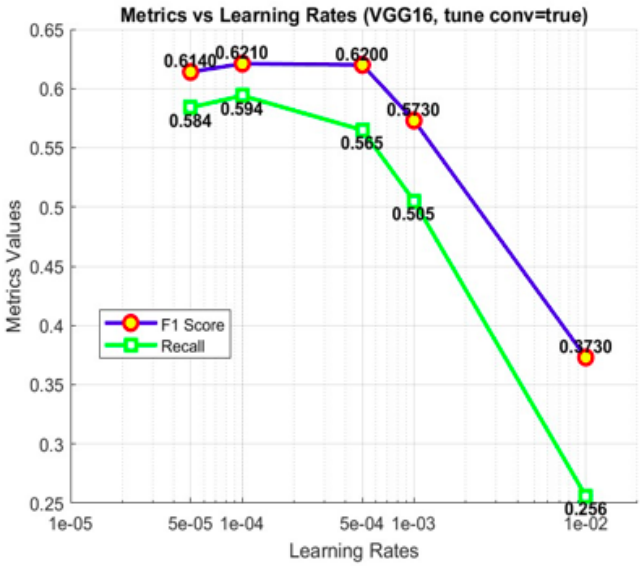


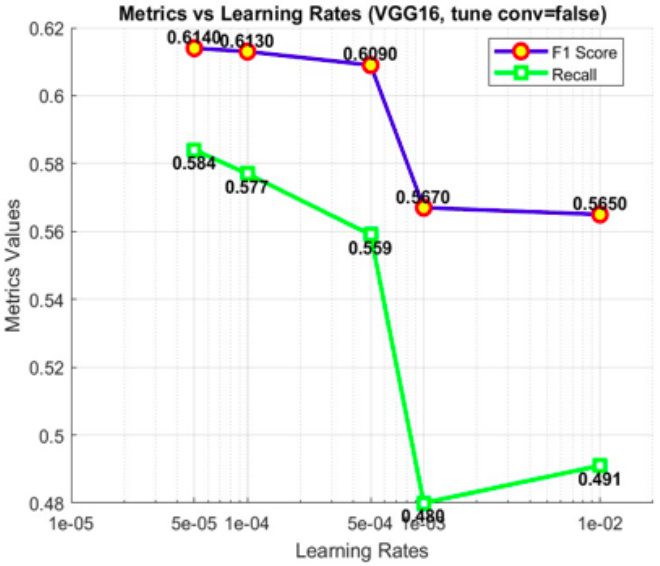


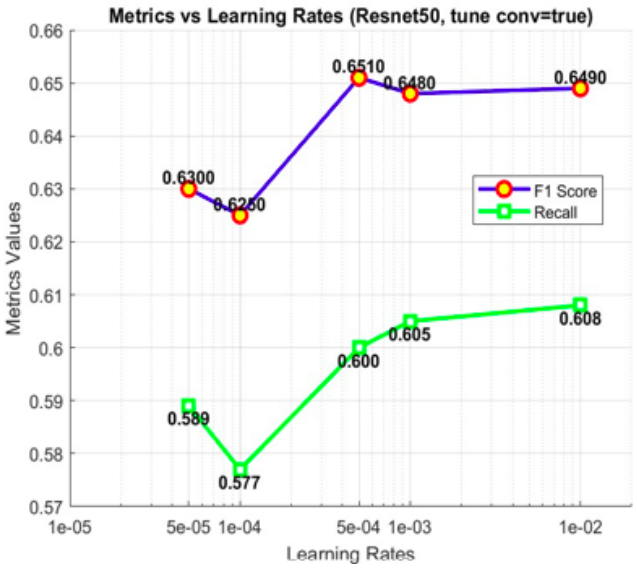


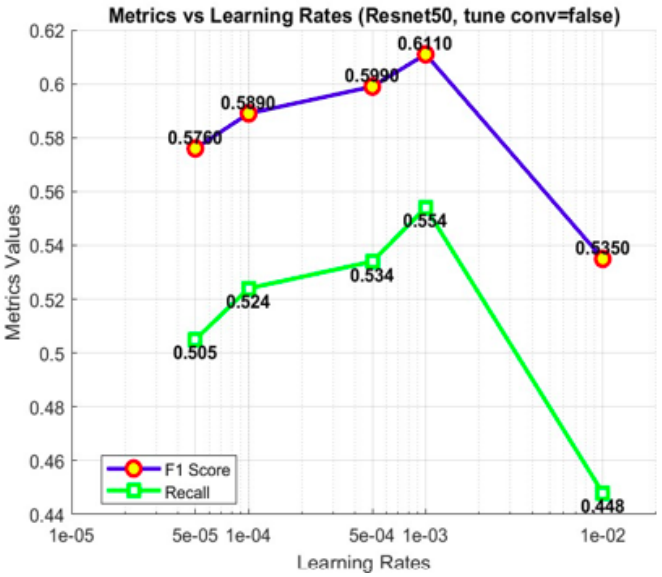
This however cannot be a direct comparison of results as we are using 10k images for our training. But the transfer learning paper helped us understand the different hyperparameters we can tune to get good results.

The primary metric we used to quantitatively gauge our studies was the **recall** . This is because when it comes to safety we need as low of false negatives as possible.









The best results from these experiments for multilabel classification is summarised below.

NN Architecture	Tune Conv	Recall	F1 Score
Resnet50	True	0.608	0.649
Resnet50	False	0.554	0.610
VGG16	True	0.594	0.620
VGG16	False	0.584	0.614

Since we were not getting good results with 224 x 224 image size (we noticed that in some of our inferences, the model failed to identify pedestrains in the scene. Especially if only a small portion of the pedestrian is present in the image). So we decided to train the model with higher resolution and chose to downsample the images to 448 by 448 pixels. Additionally since the main motive for the project is to classify vulnerable and non vulnerable obstacles, we decided to do a binary classification. We changed the categories to vulnerable(pedestrian present) and non vulnerable (no pedestrians). We compared the results between 448 x 448 and 224 x 224 pixels **image resolution**. Additionally, we changed the multi label classification to a binary classification with just two categories (vulnerable and non vulnerable).

Image Resolution	Recall	F1 Score
224 x 224	0.891	0.845
448 x 448	0.76	0.780

The last set of experiments we performed were using **learning rate scheduler**. Although this did not considerably improve the results, we got better results than than the multi label classification. Our final result was **89** recall which we consider effective within the scope of this project. Further work would be needed to make it to improve the multi label case and improve the metrics to a suitable level for deployment in real world scenarios.

Image Resolution	Tune Conv	Recall	F1 Score
224 x 224	True	0.739	0.764
224 x 224	False	0.848	0.830
448 x 448	True	0.760	0.795
448 x 448	False	0.717	0.776

Discussions

In our project, we aimed to improve the detection of vulnerable entities in images using the nuImages dataset, initially attempting a multi-label binary classification approach across 25 categories. Realizing the challenges associated with low recall values, we pivoted to a more focused binary classification problem, differentiating between “vulnerable” and “non-vulnerable” entities. Our most successful model utilized a pretrained ResNet-50, fine-tuned on 10,000 images, yielding an 89% recall and 84.5% F1-score. We learned the importance of recall in safety-critical applications, preferring false positives over false negatives to ensure no vulnerable object goes undetected. Although we successfully increased performance metrics over initial experiments, future work could explore more sophisticated neural architectures, deeper fine-tuning across more layers, and possibly integrating real-time detection capabilities.

Challenges Encountered

Our team faced several challenges throughout the project, including deciding on the optimal network architecture and managing the size and quality of training data. Early experiments revealed that neither changing the network architecture (ResNet-50 / VGGNet-16) nor significantly increasing the dataset size yielded expected performance gains. Moreover, resizing images to different scales proved less beneficial due to pretraining on standard dimensions (224x224).

If we were to start over, a different strategy would be to incorporate domain adaptation techniques to better leverage the pretraining on ImageNet for our specific task. Additionally, experimenting with more recent deep learning models or ensemble methods from the beginning could potentially improve both the efficiency and accuracy of our approach. Another consideration would be to initially focus on enhancing image resolution and quality, as this could be critical for detecting small, vulnerable objects in the dataset.
