Analysis and Improvements on E-scooter Rider Detection

Connor Brown
connorbrown@umass.edu

Daksh Dangi
ddangi@umass.edu

Apoorva Jaiswal
ajaiswal@umass.edu

1. Introduction

In the technology-forward era that we live in, travel methods are quickly leaning towards electric power. Electronic scooters, or e-scooters, are a vastly-growing method for pedestrians to transport themselves and has currently taken the United States by storm, with numerous surveys in cities like Portland, Oregon, Denver, Colorado, and Austin, Texas showcasing between 30-50% of their populations utilizing e-scooters at least once as of 2019 (Sanders et al.).

With such a boom in usage over the past few years, it only makes sense to be interested in ways of both providing safety and also better surveying, in real-time, the reality of e-scooter usage. Since the e-scooters operate on lesser speed and are not easy to spot, the riders' safety is a big concern. Being able to observe and identify e-scooters amongst immense populations of urban pedestrians would help to accomplish both of these tasks.

In this work, we analyze and improve upon a Master's thesis (Apurv) which proposed a method of utilizing two convolutional neural networks for both object detection and object localization in a synchronous pipeline to identify and isolate e-scooter riders out of a population of pedestrians. We extended this work further to also evaluate the model on dataset where pedestrians are occluded by objects. This extension was inspired by the work in Gilroy et al.. We finetuned MobileNetV2, MobileNetV3 and EfficientNet to get the model for e-scooter rider detection in both occluded and non-occluded setting. We find EfficientNet to have the highest precision of 90.6% and recall of 95.2% on e-scooter rider detection.

2. Problem statement

E-scooter rider detection is a rather new field, with two major bodies of work. Our baseline would be to utilize the dataset and approach of (Apurv). Another work in this space was done by (Gilroy et al.), which extends the idea to detect e-scooter riders in urban environments where they are prone to partial occlusions, e.g. telephone poles, other pedestrians, etc. The idea is to use the data collected in these two sets of work to both fine-tune models and then analyze the results. We then measured the accuracy, precision, recall and F1 score as well as area under the ROC curve for evaluating the performance of our fine-tuned models.

We used the dataset from (Apurv) to fine-tune the MobileNetV2, MobileNetV3 and EfficientNet models. Using the dataset in Gilroy et al., we also further finetuned EfficientNet to evaluate its performance. The first dataset includes 18,601 image segments, of which 9475 are e-scooter rider segments and 9473 are non-rider segments. The second dataset has 1130 images with 0-99% occlusion of the riders and non-riders of escooter. There are 543 e-scooter rider segments and 587 non-rider segments.

3. Related Works

3.1. E-scooter Rider Detection System

Our work stems from a thesis from Apurv, which presented a "novel vision-based system to differentiate between e-scooter riders and regular pedestrians," as well as a dataset designed to serve as a reference for models focused on building upon it, which we used. Their implementation focused on combining two state-of-the-art CNNs, YOLOv3 and MobileNetV2, to build a pipeline for both localization within larger images and, furthermore, classification of the pedestrians within as e-scooter riders or otherwise. There pipeline generated a precision of around 95% and a recall of around 75%.

Because of the fact that the dataset presented does not have the original, non-localized data, we unfortunately were unable to perform experiments on that aspect as we could not access it, so that aspect will not be discussed. Because we chose to focus on the detection section of the pipeline, we considered two different models to be used, MobileNetV3 and EfficientNet.

3.2. MobileNet

Before discussing MobileNetV3, it makes sense to first talk about its predecessor, MobileNetV2 (Sandler et al.). MobileNetV2 served to be a new mobile CNN that, at the time, improved the performance of state-of-the-art models in that realm. It is based on an "inverted residual structure," where they hold shortcut connections between thin bottle-neck layers. They also remove non-linearities within

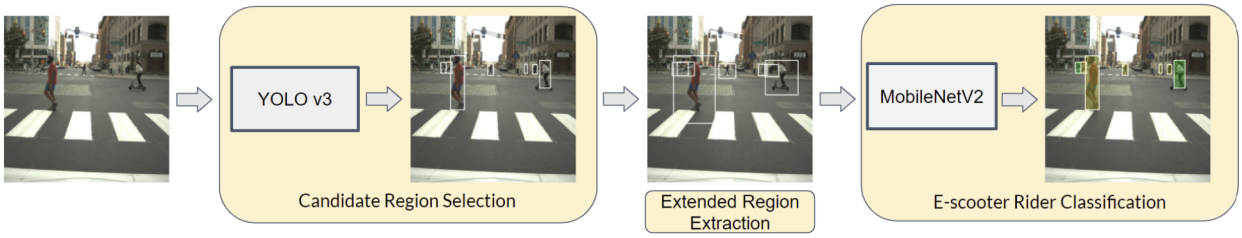

Figure 1. The Model Pipeline

these narrow layers to give the model more representational power, which served to, as demonstrated later in their proposal, vast improvements over other models in benchmarks like ImageNet classification, COCO object detection, and VOC image segmentation. This model was a huge breakthrough as it set groundwork for a family of highly-efficient models of the sort, and introducing a new type of architecture using the inverted residual blocks.

This later led to the next generation, just months after V2 was released, in the form of MobileNetV3 (Howard et al.). MobileNetV3 had a vision of being more efficient than MobileNetV2 by maintaining comparable latency on through mobile CPUs while also improving accuracy, whilst even the Large variant of MobileNetV3 reduced latency *and* improved accuracy on ImageNet classification. Because of what is presented within the paper, it is not explicitly stated that MobileNetV3 would vastly outperform MobileNetV2, but in the context of ImageNet classification, it did, which is what led us to believe it may be useful as a comparison in the e-scooter rider detection context.

3.3. EfficientNet

Finally, we consider EfficientNet (Tan and Le), released in late 2020. This model was designed to maximize efficiency of CNNs by focusing on balancing depth, width and resolution, which they found would lead to peak performance. It proposes a scaling method that can be applied to models such as ResNet or MobileNet, to reduce overall size and improve speed. When evaluated on ImageNet, their EfficientNet-B7 reached similar state-of-the-art accuracies while reducing size by a factor of 8.4, and increasing speed by a factor of 6.1, just by scaling MobileNet and ResNet models. Thus, we concluded that this could also be a valid comparison to the original thesis implementation, in the way of viewing tradeoffs as well as general efficiency.

4. Technical Approach

In the implementation (Apurv), a pipeline is proposed, utilizing "two existing state-of-the-art [as of thesis publication] convolutional neural networks (CNN), You Only Look Once (YOLOv3) and MobileNetV2." This framework serves to combine the tasks of first understanding the visual differences between a standard pedestrian, or human, and a human riding or using an e-scooter, and then applying that distinguishment to larger images with many possible classification tasks within, through localization. This results in a model that can, with good precision and recall, determine who amongst a population of many is using an e-scooter. It is important to note, however, that this thesis was published in 2021, and since then, many optimizations have been made on these models. The work in (Apurv) has created images after candidate region selection and extended region extraction. This means that we have to begin with classification of images into e-scooter rider and non-riders.

Because of this, we chose to consider the more approachable, but vastly improvable, sections that we hoped would still improve performance of the model:

4.1. Enhanced Preprocessing

The work in (Apurv) created images after candidate region selection and extension. These images were resized into the input size requirements of the MobileNetV2 model. This created sheer in the images and we look to enhance this process by adding appropriate padding to the images to create an image of NxN size. This, when downsized, to the model's input image size will now not have sheer and is expected to improve the performance.

Based on the image dimensions, we observed the mean width and height were 256 and 260 respectively. The max width and height were 2000 and 2500 respectively. Based on this, we decided to pad the images to 300 x 300 and then downsize them to 160 x 160. The results have been discussed in section 5.2.

4.2. Object Detection

We finetuned 3 pre-trained models: MobileNetV2, MobileNetV3 and EfficientNet; on data from the thesis (Apurv). After this, we chose EfficientNet to be fine-tuned on the data from Gilroy et al. to compare its performance on images with and without e-scooter riders that have been

Figure 2. Pedestrian with an e-scooter, with 85% occlusion

occluded. We fine-tuned these models by modifying the last fully-connected layer for classification for MobileNetV2 and EfficientNet but we added an extra layer with 2 input neurons and 2 output neurons for MobileNetV3. We were faced with hardware limitations as we tried fine-tuning our model using the entire dataset, so we instead opted to train on a smaller, more manageable sample to produce some preliminary results for this milestone. In addition, we also made the choice, collectively, to implement this project mostly using PyTorch rather than TensorFlow. The thesis implementation utilized TensorFlow, and we decided to move away from it for the sake of exposure, and to hone skills with the relevant frameworks.

4.3. Tuning on Occluded Images

We decided to fine-tune our models further using occluded data, where riders or pedestrians have visible obstructions preventing them from being easily identifiable (from no fault of their own). An example of one of these instances is shown in Figure 6. These environmental impacts make it difficult for a model to conclude whether a person has an e-scooter or not consistently, so we believe that allowing the model to become familiar with them could prove beneficial for the model long-term. It is important to note, however, that these occlusions (the objects that cover the rider or e-scooter in a given image) were post-processed, or edited into the picture. As such, they may not accurately simulate an environment conducive to occlusions in the real world, and is something to taking into consideration during evaluation.

5. Evaluation

5.1. Training and Fine-tuning

Our first steps involved procuring publicly-available, pre-trained weights for MobileNetV2, MobileNetV3 and EfficientNet. After this, we set up our models for the ex-

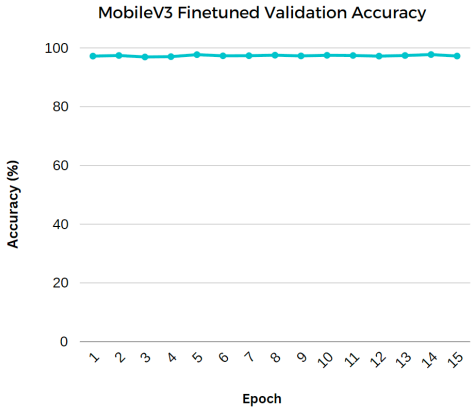

Figure 3. Finetuned validation accuracy over Epochs for MobileNetV3

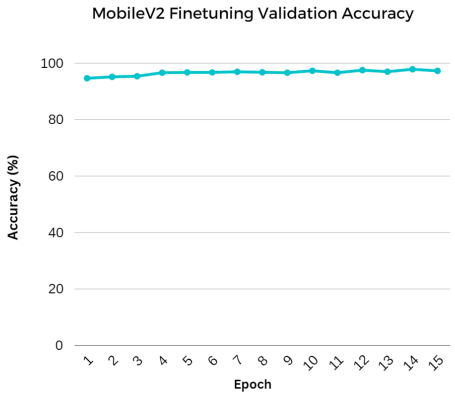

Figure 4. Finetuned validation accuracy over Epochs for MobileNetV2

periments we planned on, which involved adding a global-pooling layer to the end of MobileNetV2's architecture, before the classifier layer, and also changing its output features to size of 2, to reflect our classification task. MobileNetV3 already had this pooling layer, however, so we opted to not add one. Its classifier layer also had two fully-connected layers, one that up-scaled its input from 960 features to 1280, and one that downscaled it from 1280 to 1000. Since our task requires output of 2, we felt that upscaling could hurt information gained from the network, so we replaced the upscaling with downscaling, going from 960 features to 480 features, and then finally 480 to 2. EfficientNet only required us to change the output features to 2.

After this, we began fine-tuning the models, by first freezing all but their classifier layers, and then iterating 10 epochs each over the subset of the original training set, producing intermediate validation accuracies as the process

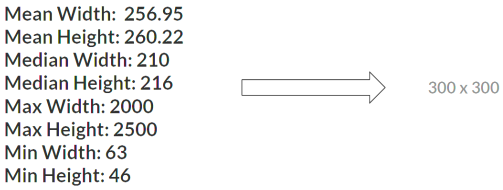

Figure 5. Process of Padding

went on. After this, we unfroze the last 26 layers right before the classifier, to produce almost a gradient of information towards our task, and then ran another 15 epochs on the training subset. This produced models without any change in pre-processing or additions of occluded data. Finetuned validation accuracy graphs are shown for MobileNetV3 and MobileNetV2 in Figures 3 and 4, respectively. As can be seen, both models converge to around the same validation accuracy by the end, around 97%, but it is also worth noting that MobileNetV3 starts around that accuracy to begin with, while MobileNetV2 takes some time to catch up. Therefore, it can be inferred that, out of the box, MobileNetV3 performs better than MobileNetV2, which makes sense considering its goals as a model.

5.2. Pre-processing Changes

We then decided to try improving upon the pre-processing steps of the data by accounting for sheer. The process (shown in Figure 5) involved padding all images to size 300x300, and *then* resizing to 160x160, instead of just squishing images instantly to 300x300. We believed this would help reduce sheer from the resizing process, potentially improving our data's validity in this domain.

After doing so, we then fine-tuned both MobileNetV3 and EfficientNet, on that data, in the same way as we did in the previous section, running 10 epochs and then 15 epochs, with the same sets of frozen layers.

5.3. Fine-tuning on Occluded Data

Following this, we began working with the occluded dataset we obtained from Gilroy et al.. Similarly to before, we added a new fully connected layer which downscaled the incoming features to 2 for the purpose of our classifier. To accomplish our task of fine-tuning over the occluded data, we once again used an Adam optimizer to iterate over 10 epochs using the train/val split initialized for the previous task to tune the newly added full-connected layer, while freezing all of the other layers simultaneously.

From here, however, we switched to a train/validation split over the occluded data. We also added another fully connected layer which mapped 2 input channels to 2 output channels. We hoped that this change in architecture would capture the occlusions from the new dataset it was being trained over, and make more accurate predictions in the case of occlusions in the test set. The model was then ran for another 10 epochs in order for the final fully-connected layer to learn it's weights. MobileNetV2 was able to achieve a validation accuracy of only 59%, while MobileNetV3 was able to achieve 69% over the occluded data. We attribute the low validation accuracy to the limited dataset we had to train our model over.

The models were then fine-tuned over the test/validation split we had created using the dataset from Apurv for another 15 epochs after unfreezing the last 26 layers - mimicking the procedure followed in Apurv's thesis. MobileNetV2 was observed to have a peak validation accuracy of 97.60%, and MobileNetV3 had a peak validation accuracy of 98.43%.

5.4. Test Accuracies

For a data split, we used a 70-15-15 split on the original training data, to reduce the size of our data for resource reasons. For the process of evaluation, we ran the models, at first, through the test data that we subsetted from the original training split, but found that the process of evaluation actually showed that our model was overfitting that space, with test accuracies pushing 99.6%. After some thought, we began to understand that we were essentially training, validating and testing all on a small piece of the overall domain, and decided to instead maintain our prior training and validation method, but test on the unseen original test split from the dataset the thesis provided us. This showed far more reasonable results, as shown in the next section. We followed this process for every model tested above, producing test accuracies, precision, recall, F1 and ROC Curve graphs, to see the relationship between true positive rate and false positive rate.

6. Results

6.1. Tasks and Setbacks

We performed the following set of tasks:

1. Attempted enhancing of image quality with padding
2. Finetuning MobileNetV2 model
3. Finetuning MobileNetV3 model
4. Finetuning EfficientNet model
5. Finetuning EfficientNet with pre-processed images
6. Finetuning all the models with 0-99% occlusion based images
7. Comparing results and analysing the classification capabilities of the architectures

When we started the work of implementing the thesis by Apurv, we identified that the region selection and extension tasks were already executed as part of the work, although this has not been mentioned in the paper. This resulted in a pivot and we requested Gilroy et al. for the dataset used

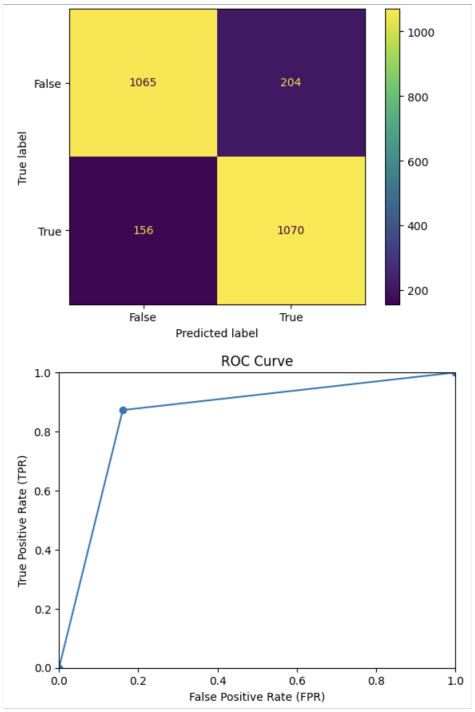

Figure 6. MobileNetV3 results

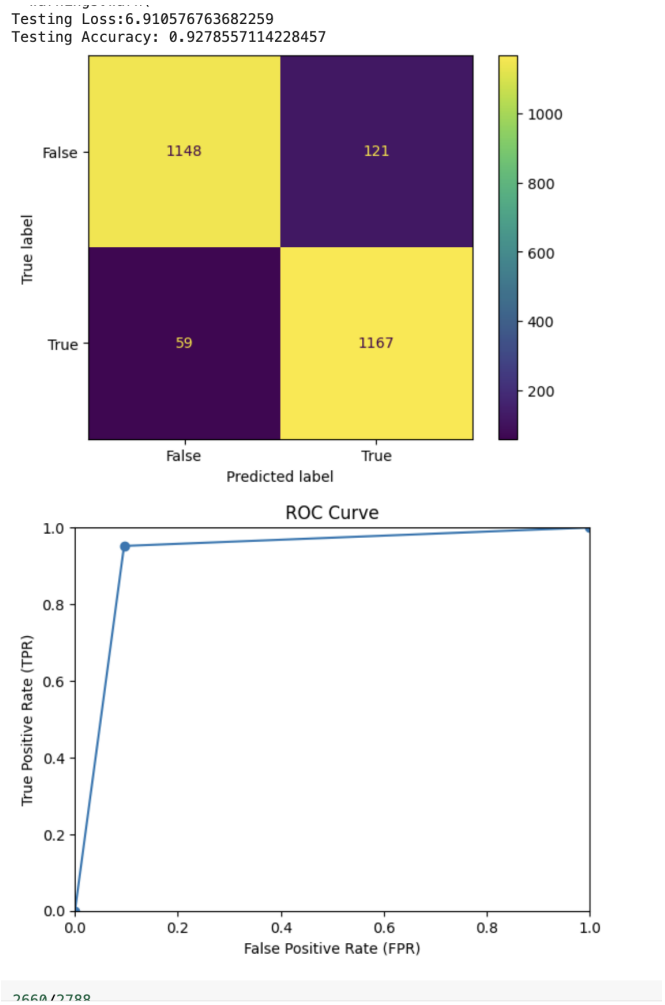

Figure 7. EfficientNet Results

by their team to fine-tune a model that had enhanced detection capabilities on an occluded dataset. Currently, we have code that uses pretrained, publicly-available weights of MobileNetV2 and fine-tunes a new, fully-connected layer and an output layer for classification with other layers frozen.

Since training the models with the datasets was taking very long, we ended up using the training dataset of size 18601 by Apurv and divided that into training, test and validation set with a 0.70,0.15,0.15 split ratio. This enabled us to run the training and our results are shown as follows.

6.2. Metrics

As can be seen in 9, our results are quite surprising across the board. MobileNetV2 performed about as well as it did in the thesis, producing notably high recall and the third highest accuracy for all of our models. MobileNetV3, however, fell short in almost all metrics, which surprised us. EfficientNet, however, being the overall best model from our experiments, boasted the highest accuracy, F1 and precision, with still great recall.

It is definitely worth noting that EfficientNet's finetuning on padding images actually hurt the model, seeing lowered metrics across the board and hurting recall, implying that it had a lot of false negatives, which is exactly what we don't want from a model that is designed for safety precautions. Our reasoning for this observation has do with the dataset we used - specifically the large difference in width/height amongst the training/testing images, which could have decreased the usefullness of padding.

As for the occluded finetuning, we did see some interesting results. Recall was improved in for MobileNetv3, and reduced slightly for EfficientNet and MobileNetV2, but did not produce better accuracies or precision for any of the 3 models. This is unfortunate, as we had hoped these results would be more beneficial to the domain.

Examples of the ROC Curves we produced as well as confusion matrices are showing in Figures 6, 8 and 7.

6.3. Analysis of Metrics

From the metrics, we first noticed the lower performance from MobileNetV3. This can be most likely attributed to the original idea behind MobileNet models, which is to

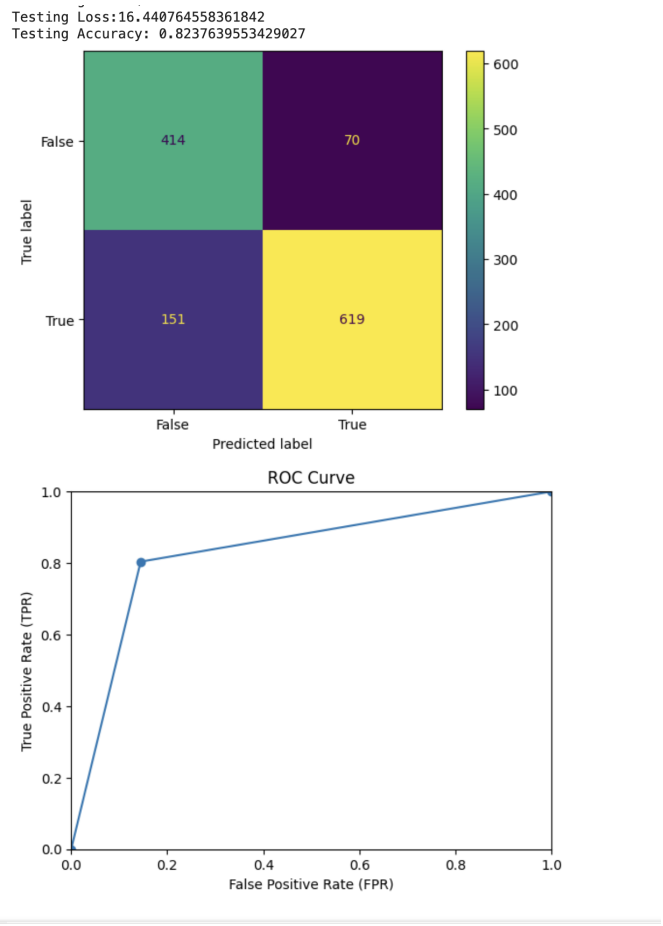

Figure 8. EfficientNet with Padding Results

function off of mobile CPUs with low latency, while still boasting state-of-the-art results. MobileNetV2 did change the scene with its implementation, but its successor was actually designed with efficiency in mind, not necessarily performance. In fact, MobileNetV3 was only noted to improve performance in some benchmarks, and only at a small margin. Thus, it can be concluded that, in this domain, MobileNetV3 simply does not succeed MobileNetV2, and could be a key reason why the original authors chose it.

However, EfficientNet's performance is very surprising, as it was designed to improve efficiency by downscaling the size of models while retaining performance. We believe that the reduction of the model's size may have produced a less complex model, allowing it to adhere better to generalization over the dataset, leading to better results in evaluation.

For our pre-processing attempts, the reduced metrics can be potentially attributed to the fact that not all images are the same size, leading to some images potentially being harmed by the addition of padding, leading to the images being harder to classify correctly.

Model	Precision	Recall	Accuracy	F1-score
MobileNetV2	83.8%	97.8%	89.6%	90.3%
MobileNetV3	84.0%	87.3%	85.6%	85.6%
EfficientNet	**90.6%**	**95.2%**	**92.8%**	**92.8%**
EfficientNet with padded images	89.8%	80.4%	82.4%	84.8%
MobileNetV2 - occluded dataset	66.7%	96.4%	74.6%	78.9%
MobileNetV3 - occluded dataset	73.8%	94.3%	80.8%	82.8%
EfficientNet - occluded dataset	85.8%	94.6%	89.7%	90.0%

Figure 9. Metrics for Each Model Tested

Finally, our occluded finetuning results are noticeably lower across the board, but it is very good to see such high recall. This high recall shows that training on occluded data makes the model less likely to classify an image has not having an e-scooter, which would imply that a camera using this model is less likely to make a potentially endangering call on a pedestrian.

7. Conclusion

The results of our experiments show that EfficientNet can potentially serve as a better and, as the name implies, more efficient model for the task of classifying e-scooters in the pedestrian domain. This is illustrated through not only it's high accuracy, but more importantly, it's low false positive rate, which is arguably one of the more important metrics when considering a model designed for safety precautions. This is especially useful in a domain that focuses so heavily on precaution when it comes to things like self-driving cars. We were especially surprised by this model's performance, but even more so by the lack of improvement MobileNetV2's successor had.

The results of our experiments on pre-processing and occluded images also showed a lack of improvements, but did show that occluded data can be responsible for improving recall, which is something to be further explored. We would consider other methods of applying the data, such as augmentation rather than fine-tuning, and also seeing how the model could fare when training on occluded data itself. As for pre-processing, we still believe there can be improvements to the data to normalize their sizes to better reflect the domain, as many images are tall or wide, and are not very similar to each other. Thus, it may be worth continuing to explore the idea of padding or something similar to remove sheer within the data.

This project proved to be a great experience with PyTorch as well as applying different models to similar processes to push a domain that is quite unexplored but still very beneficial to the public, and we are especially looking forward to seeing what further improvements are brought to the problem of keeping e-scooter riders safe.

References

[1] Kumar Apurv. E-scooter rider detection system in driving environments, 2021. 1, 2, 4, 5

[2] Shane Gilroy, Darragh Mullins, Edward Jones, Ashkan Parsi, and Martin Glavin. E-scooter rider detection and classification in dense urban environments. *Results in Engineering*, 16: 100677, 2022. 1, 2, 4

[3] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, Quoc V. Le, and Hartwig Adam. Searching for mobilenetv3. *CoRR*, abs/1905.02244, 2019. 2

[4] Rebecca L. Sanders, Michael Branion-Calles, and Trisalyn A. Nelson. To scoot or not to scoot: Findings from a recent survey about the benefits and barriers of using e-scooters for riders and non-riders. *Transportation Research Part A: Policy and Practice*, 139:217–227, 2020. 1

[5] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. 1

[6] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020. 2