Drowsiness Detection System

By: Kierra Dangerfield

Table of Contents

I. Introduction

Each year, drowsy driving accounts for about 100,000 crashes, 71,000 injuries and 1,550 fatalities, according to the National Safety Council (NSC). In addition, a study from the AAA Foundation for Traffic Safety found that drowsiness was a contributing factor in up to 9.5 percent of all crashes and 10.8 percent of crashes that included airbag deployment, injury or significant property damage.

Car companies can develop a drowsiness detection in their vehicles to alert drivers when they start to fall asleep at the wheel.

My goal is to create one deep learning model that predicts if a person is yawning or not and another one that predicts if an eye is closed or open based on an image dataset. When combined, these elements could help form a more robust drowsiness detection model.

II. Data Overview

The data set came from Kaggle. It is noted that I am unsure of how the data was originally obtained, but I did manually view the images of each class. It is already split between a testing and training set. The classes are Closed, Open, yawn, and no_yawn. Originally, there were 2,900 images in the dataset. Each directory has a four classes:

- **Closed** - up close image of the eye closed
- **Open** - up close image of the eye open
- **Yawn** - drivers in a car yawning
- **No_yawn** - drivers in a car no yawning

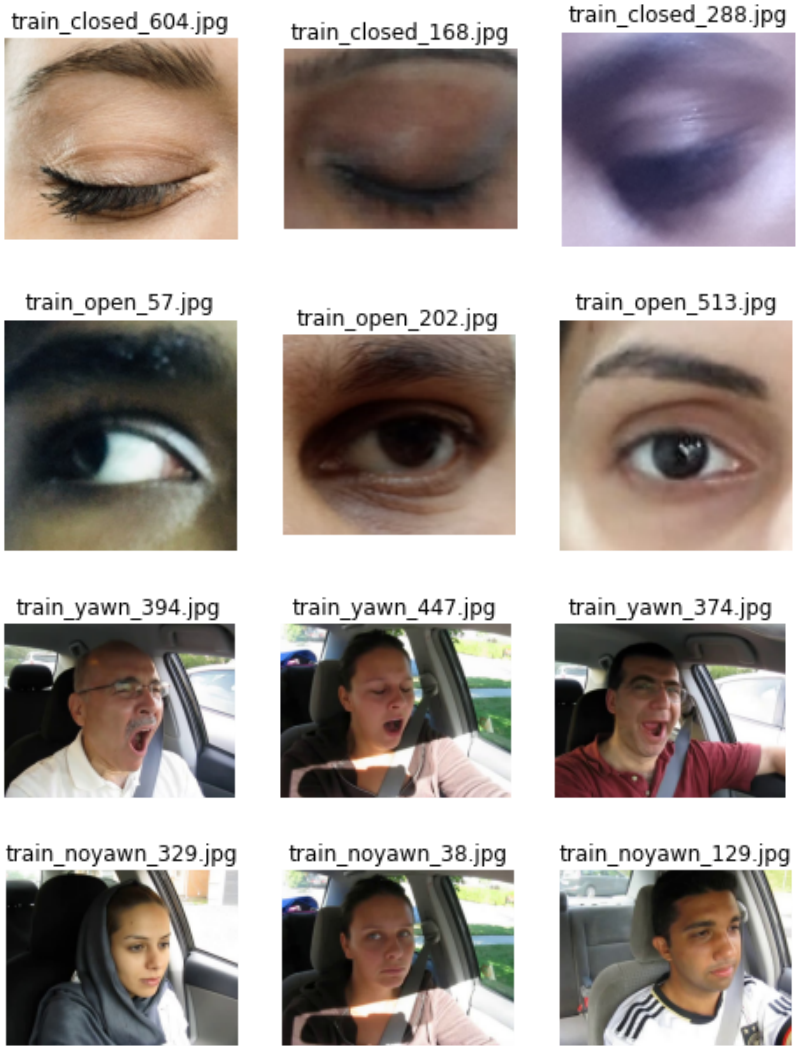
The Closed train set has 627 images and 109 images in the test set. The Open train set has 617 images and 109 images in the test set. The yawn train set has 637 images and

106 images in the test set. The no_yawn train set has 627 images and 109 images in the test set.

It is important to note that there is a difference in the way that the images were captured between the yawn and no_yawn classes. The no_yawn class does not have pictures that show the steering wheel. The yawn class has images of drivers that

EDA

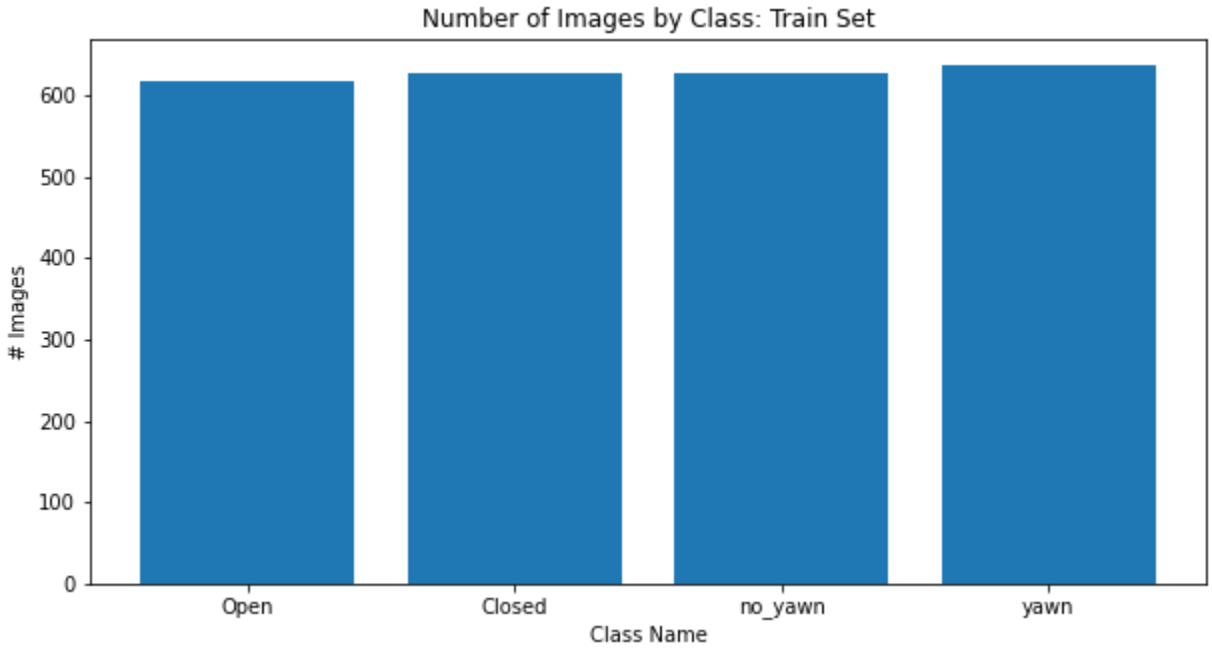
A. Random Images



Above are random images from the training set from each class. The closed and open images are zoomed in compared to the yawning and no

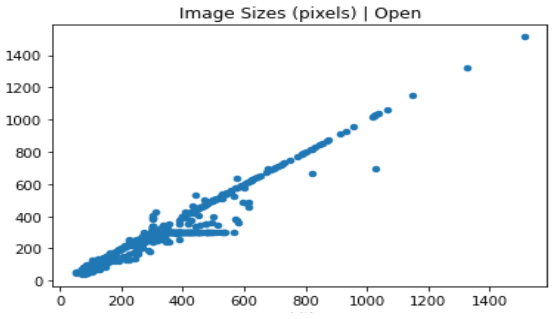
yawning images. Unlike the closed and open images, the yawning and no yawning images are showing a person in a car and are not zoomed in.

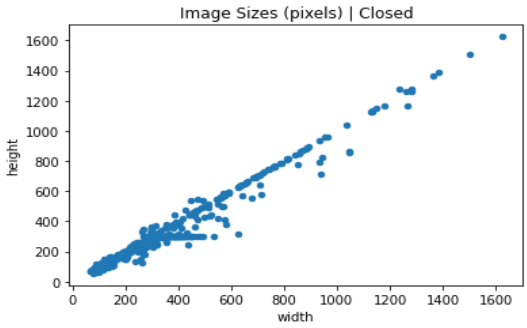
B. Class Size

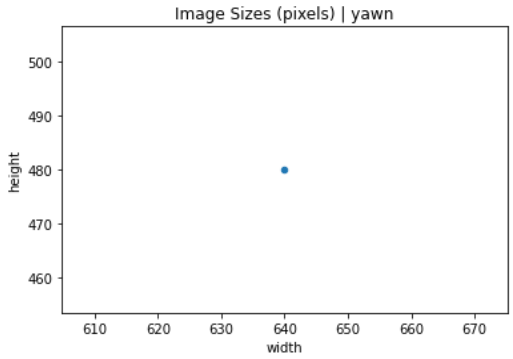


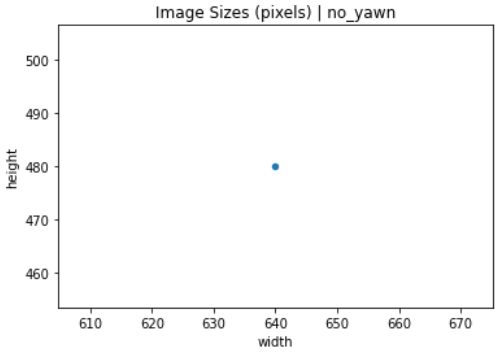
The above is showing the number of images in each class of the training set. It shows that the classes have a similar number of images. Open has 617 images. Closed has 627 images. No_yawn has 627 images. Yawn has 637 images.

C. Image Size







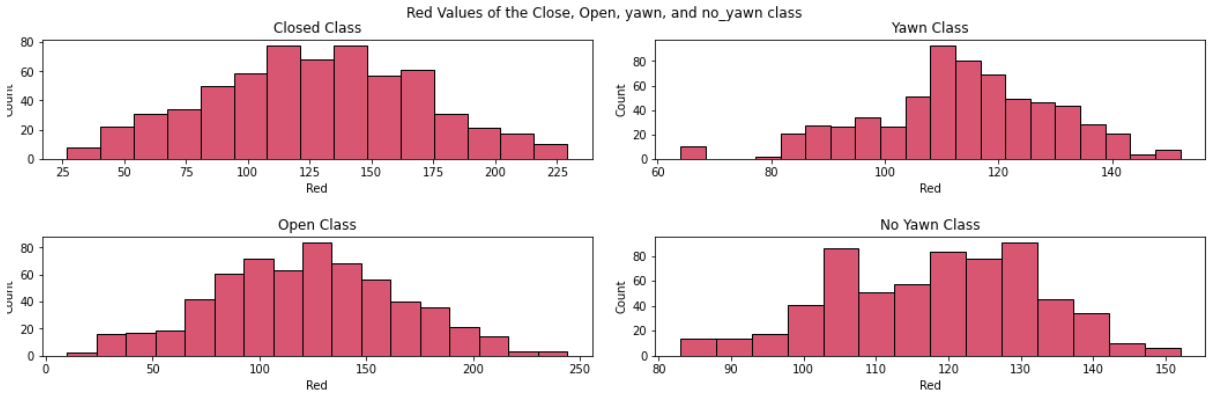


The above are the image sizes of each class of the training set. The yawn and no_yawn classes all have the same image size. The open and closed classes have a variety of image sizes. Those classes will need to be resized for the model.

D. RGB of train set

Below charts are showing the average red, green, and blue channel values of each image in each class.

Red

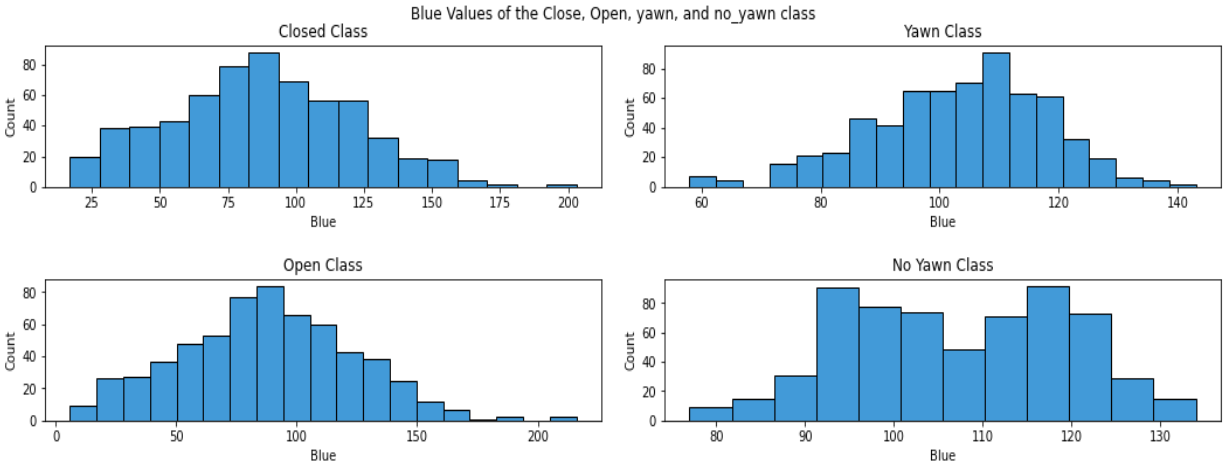


Above is the distribution of the average red channels of each class. Between the yawn and no_yawn classes, the no yawn class has a slightly higher average of red values compared to the yawn class. The open class has a higher distribution of the average red channels compared to the closed class.

Green

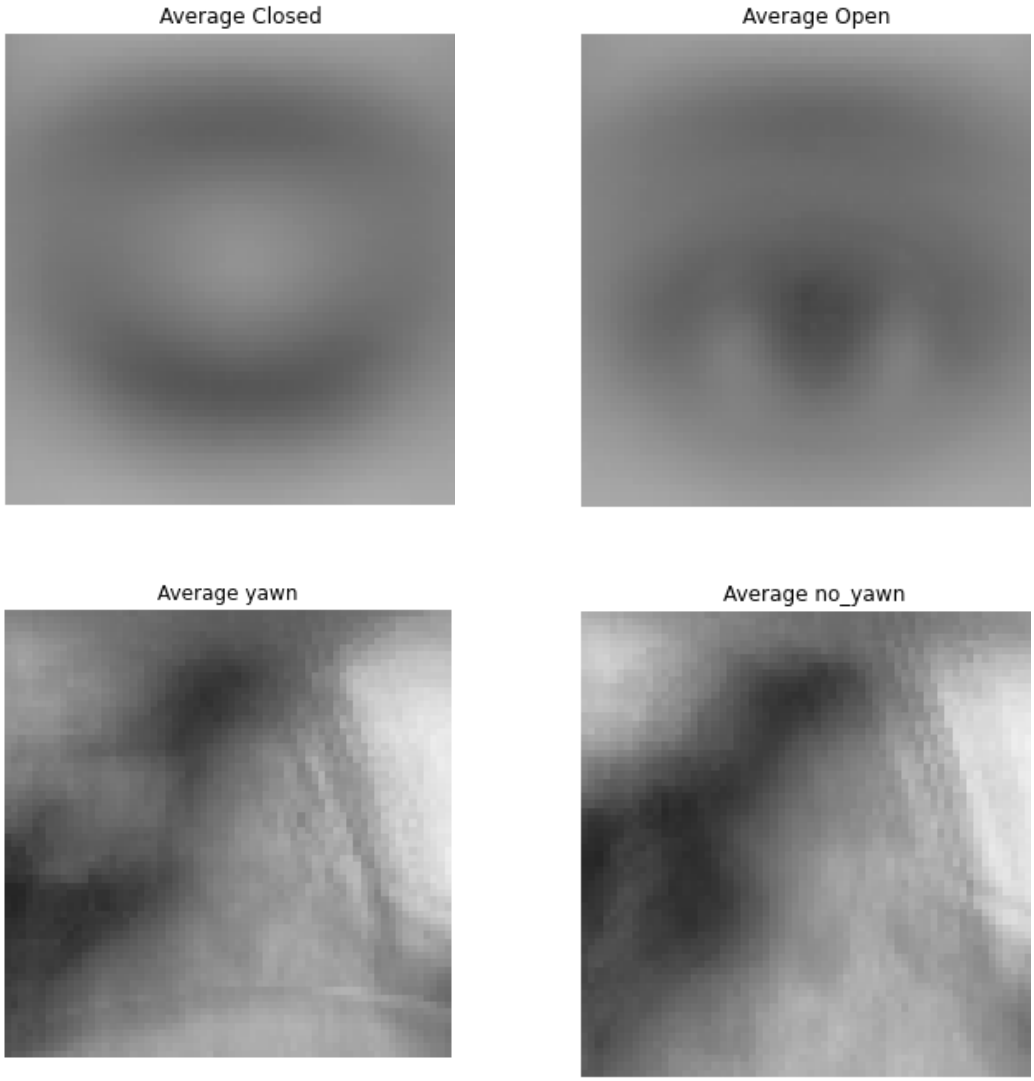
Above is the distribution of the average green channels of each class. Between the yawn and no_yawn classes, the yawn class has a slightly higher average of green values compared to the no yawn class. In the yawn class, there is a bar that represents a higher frequency compared to the other bars for the average green channels of the images. The closed class and the open class have similar distributions of the average mean of green channels.

Blue



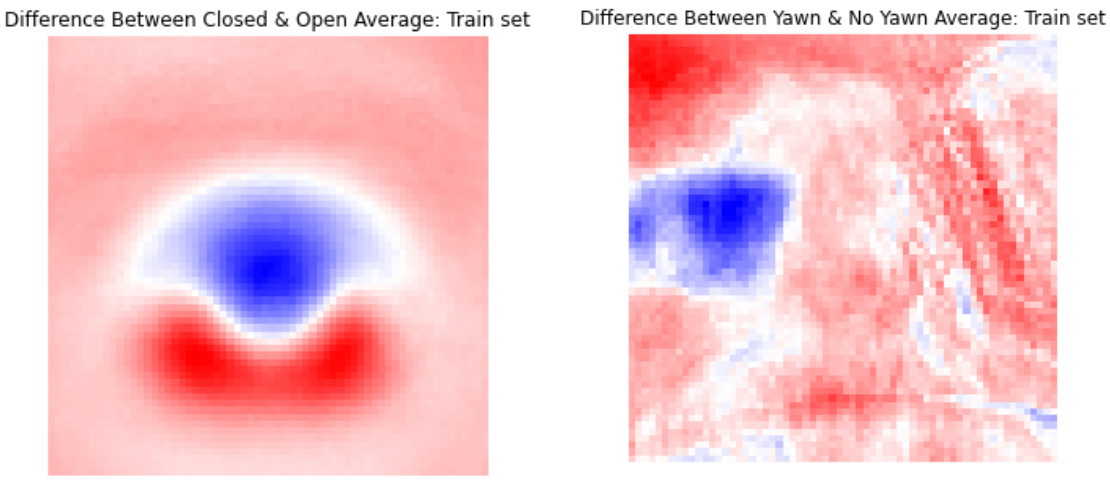
Between the yawn and no_yawn classes, the yawn class has a slightly higher distribution of average blue channel values compared to the no yawn class. The closed and open classes have similar distributions of the average blue values.

C. Average Image for Each Class



The above are the average images of each class of the training set. For the closed class, the average closed eye shows the eyelid, eyebrow, and eyelashes. For the open class, the average open eye shows the iris and the brow relatively cogently so it does appear these two have consistent differences. For the average yawn and no_yawn classes, it is harder to differentiate between the two. This likely indicates this will be more difficult to predict.

D. Contrast

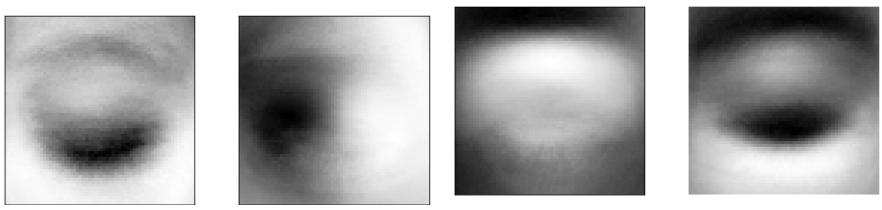


The above shows the difference between the open and closed classes and the yawn and no yawn classes in the training set. The blue is showing the areas that are different based on the average image. I can see where the difference is between the closed and open classes, as the open eye and iris particularly pop out . The yawn and no_yawn classes seem to mainly focus on the background of the average images.

E. Eigenimages

Below are Eigenimages of each class in the training set. These eigenimages give us an idea of what the different important aspects of the image are that make up the image as a whole. These can be indicative of potentially important features.

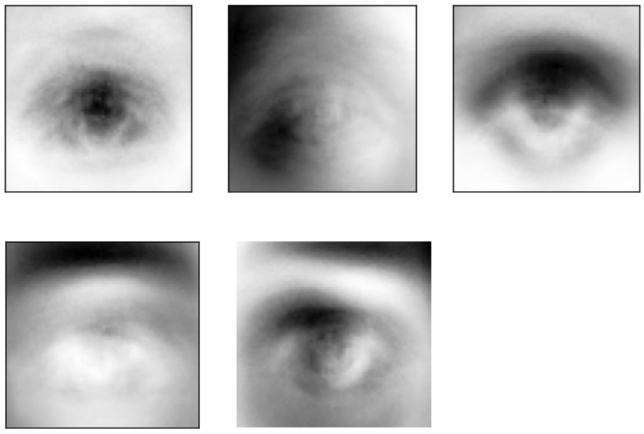
Closed



For the closed eye, we can see rather distinctive features. The first image is focusing on the eyelashes and bottom of the eye. The second image seems to relate to the shadow detailing the curve of the eye from left to right, while the

fourth image does the same for the curve up to down. The third image seems to separate out the area where the eye is from the rest of the picture.

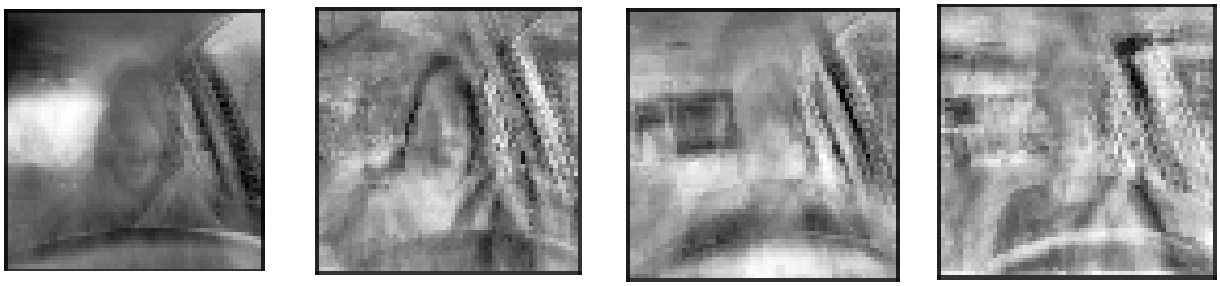
Open



For the Open class, the first image is focusing on the cornea and iris of the eye. The second image is focusing on the curve of the eye from left to right. The third image is focusing on the eyebrow structure and the way the eye sits under the eyebrow. The fourth image seems, similarly to the third image from the closed class, to focus on separating the area where the eye is from the background. The fifth image seems to focus on the shadow of the nose and the structures around the eye.

Yawn

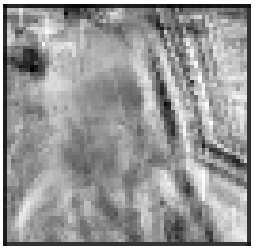
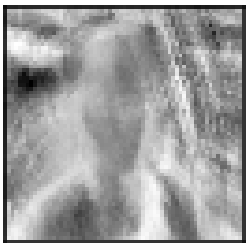
For the yawn and no yawn classes, it is harder to distinguish between the images. There were 17 eigenvectors for yawn and 17 for no-yawn. In each case I chose a few of these features that seemed to be more recognizable in what they represented.



For the yawn class, the first image there is a focus on the back window versus the foreground of the car. The second image is focusing on the face of the driver sitting in the car. The third image is focusing on elements on the inside of the car. The fourth image is focusing on the body of the driver's positioning. Of these it seems likely that the second eigenvector represented here would be closest to a useful feature for predicting yawning.

No yawn



For the no yawn class, the first image is focusing on the inside of the car. The second image is focusing on the face of the driver. The third image is focusing on the window of the car. In the fourth image, there is a focus on the body of the driver.

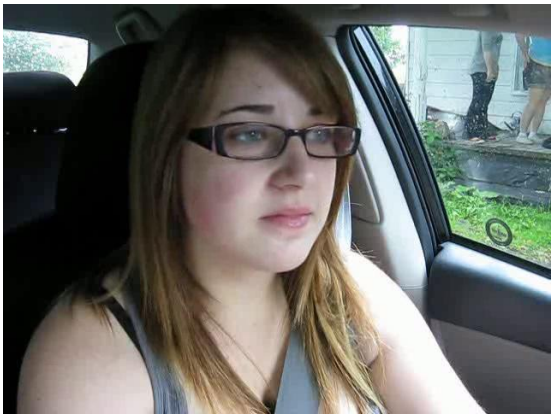
III. Image Preprocessing

Resized

As discovered before, the Closed and Open classes are different sizes, so they need to be resized. I put the resized images in their own folder. The resized images are 300 by 300 pixels.

Grayscale

I gray-scaled the images of each set of classes and put them in their own folder using OpenCV. Below are some random images from each class.




IV. Modeling

I began by separating the models between the yawn/ No yawn class and closed/open classes. For the yawn classifier, I chose “yawn” as the positive class as from a business perspective we will care more about the recall and precision of predicting yawning than not-yawning, since it’s important to make sure we know when a driver is putting themselves or others at risk. For the same reason, I chose “closed”, as the positive class for training the closed/open classifier. For both models I created a Dummy Classifier, Sequential, and a pretrained model called MobileNetV2. The yawn and no_yawn classes have a few more images in the training set compared to the Closed and Open classes. Because of this I was able to standardize the Closed and Open classes for the model but not the yawn and no_yawn classes. The Closed and Open classes model had an epoch of 15 with a batch size of 32. The yawn and no_yawn classes model had an epoch of 10 with a batch size of 32. Below are the model results for the classes using the training set and validation set.

Model	loss	accuracy	val_loss	val_accuracy
Open/Closed: Sequential	0.0338	0.9925	0.4384	0.9016
Open/Closed: MobileNetV2	0.0853	0.9784	1.2524	0.4980

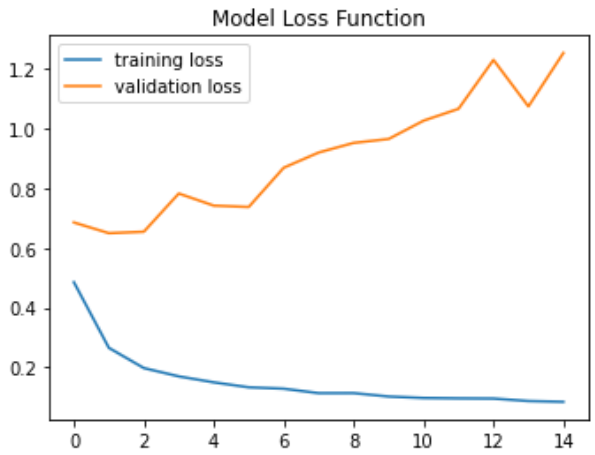
Below are the model metrics.

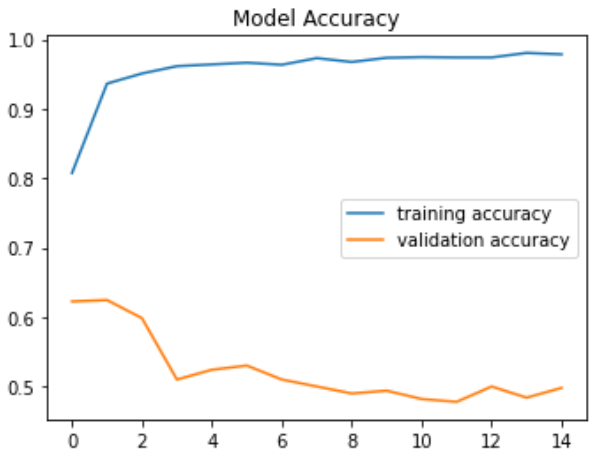
Model	AUC	Precision	Recall	val_auc	val_precision	val_recall
Open/Closed: Sequential	0.9996	0.9925	0.9925	0.9470	0.9047	0.8956
Open/Closed: MobileNetV2	0.9962	0.9764	0.9764	0.5797	0.5000	0.4940

The dummy classifier simply predicts the most common label.
Given the dataset is balanced, this results in the Open and Closed classes having an accuracy of 0.50 for the training set. So we can see that our trained models significantly outperform it.

Below is a graph depicting the model loss and accuracy for the Open and Closed classes for the sequential model.

Below is a graph depicting the model loss and accuracy for the Open and Closed classes for the pretrained model using MobileNetV2.





The sequential model performed better than the MobileNetV2 model for the Open and Closed classes.

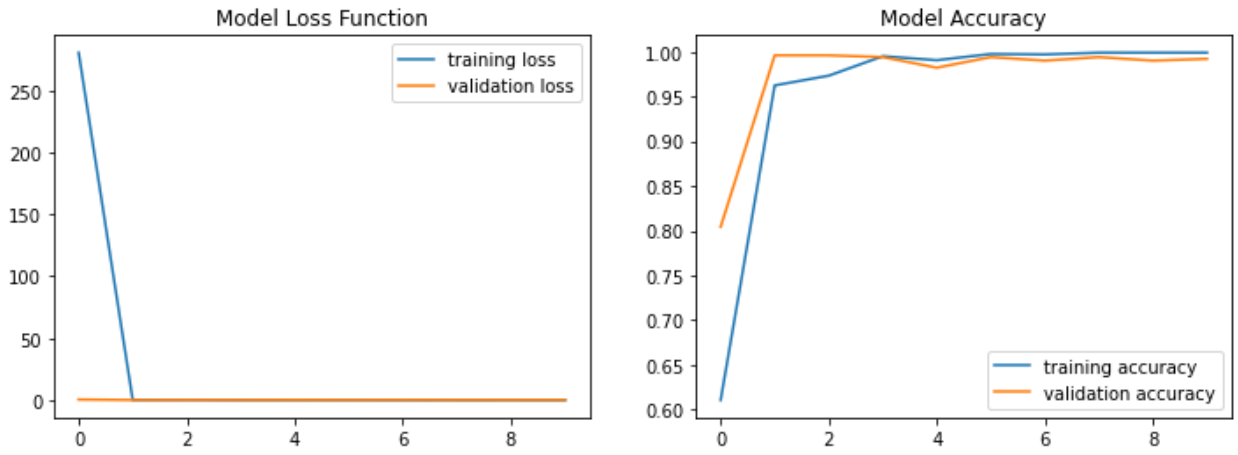
Model	loss	accuracy	val_loss	val_accuracy
yawn/no_yawn: Sequential	0.0038	0.9990	0.1181	0.9921
yawn/no_yawn: MobileNetV2	0.0406	0.9955	0.0377	0.9960

Below are the model metrics.

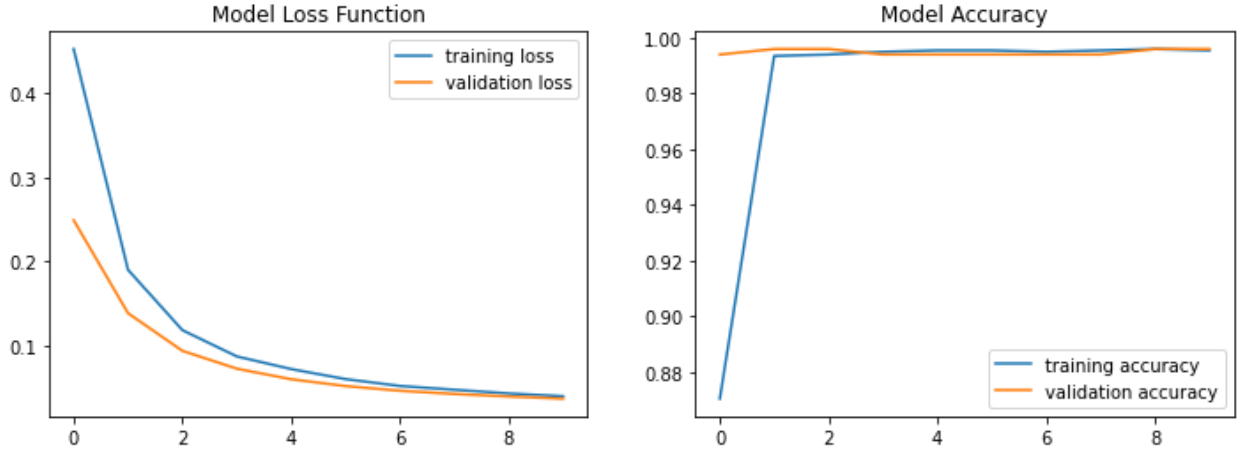
Model	AUC	Precision	Recall	val_auc	val_precision	val_recall
yawn/no_yawn: Sequential	0.9997	0.9990	0.9990	0.9947	0.9941	0.9921
yawn/no_yawn: MobileNetV2	0.9977	0.9955	0.9955	0.9974	0.9941	0.9960

Again, given that the classes are balanced, the Dummy Classifier has a score of 0.50 for the training set, which both models significantly outperform.

Below is a graph depicting the model loss and accuracy for the yawn and no_yawn classes for the sequential model.



Below is a graph depicting the model loss and accuracy for the yawn and no_yawn classes for the pretrained model using MobileNetV2.



The MobileNetV2 model performed better than the sequential model for theyawn and no_yawn classes.

Threshold Moving

These models can be used by car companies to alert drivers if they are yawning or closing their eyes too much. If the models predict a false positive, the driver would be alerted that they are dozing off when they are not. This could cause the driver to be annoyed and not want to purchase a car with this feature.

If the models predict a false negative, it would predict that the driver isn't yawning or has their eyes open even though they are yawning and their eyes are closed. This can result in a feature that is not reliable and could present a real danger to the driver and others. Therefore, false negatives seem to be the worse of the two types of errors in the scenario.

To improve this model we can optimize the Recall score because there's an importance of both precision and recall. We can improve the model by moving the threshold. Below are the current metrics based on the threshold of 0.50 on the test set.

Model	loss	accuracy	AUC	Precision	Recall
Open/Closed: Sequential	0.0249	0.9908	0.9997	0.9908	0.9908
yawn/no_yawn: MobileNetV2	0.0502	0.9930	0.9929	0.9930	0.9930

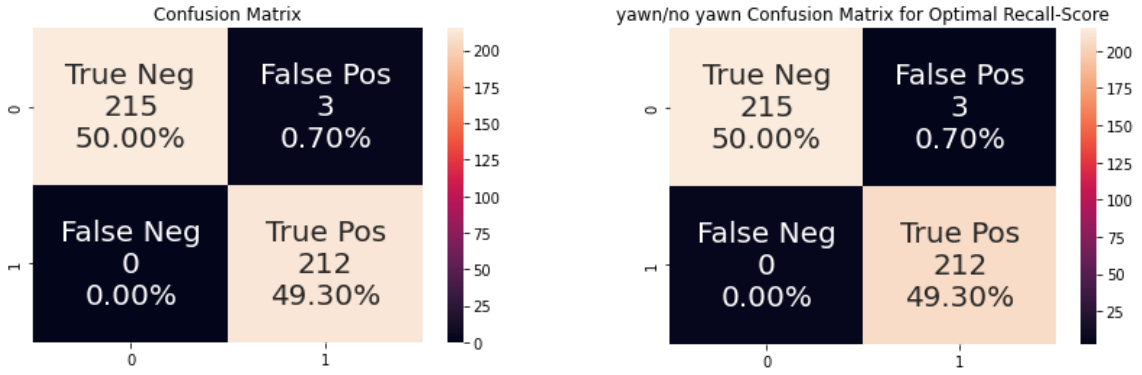
For the yawn and no_yawn model with MobileNetV2, below is the confusion matrix with the current threshold of 0.50 and the confusion matrix with the optimal recall score. An optimized recall score would mean that the threshold would be at 0.00. However a model with a 0 threshold would be an unreliable model because everything can't be labeled as yawning. I made the threshold 0.15.

Below is a table showing the model metrics between the threshold of 0.50 and 0.15 for the predictions of the test set for yawn/no_yawn.

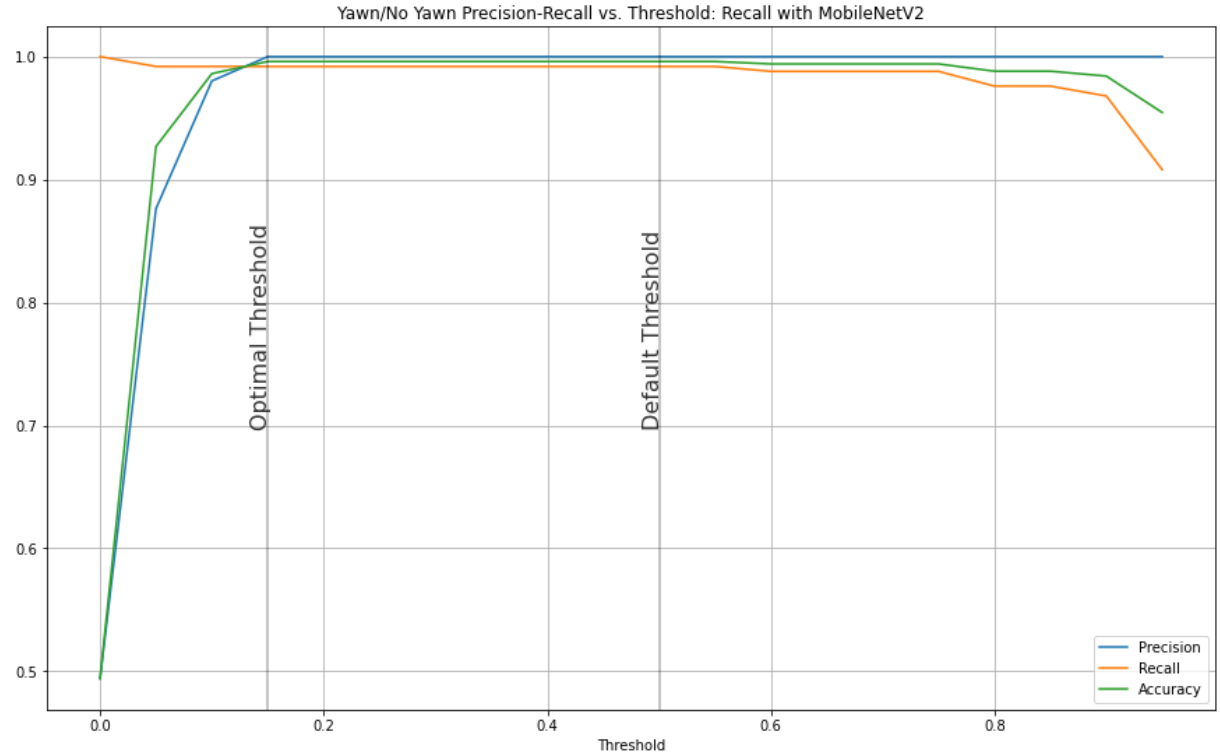
Model	Recall	Precision	Accuracy
yawn/no_yawn with 0.50 threshold	1.00	0.98605	0.99302
yawn/no_yawn with 0.15 threshold	0.97170	0.98565	0.97907

As you can see this particular model didn't need the threshold to move because it was already 1.0 for the test set, and all of the other metrics are higher at the 0.50 threshold mark.

Below is the confusion matrix of the model at 0.50 and 0.15 threshold. As you can see there weren't any changes.



Below is a precision-recall vs threshold chart showing the default threshold and the optimized threshold with the model metrics

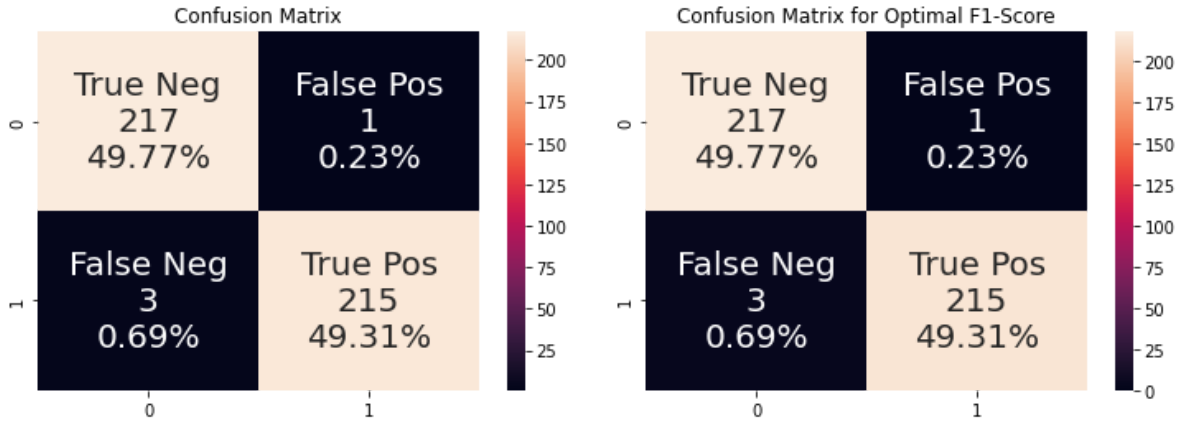


Below is a table showing the model metrics between the threshold of 0.50 and 0.15 for the predictions of the test set for closed/open.

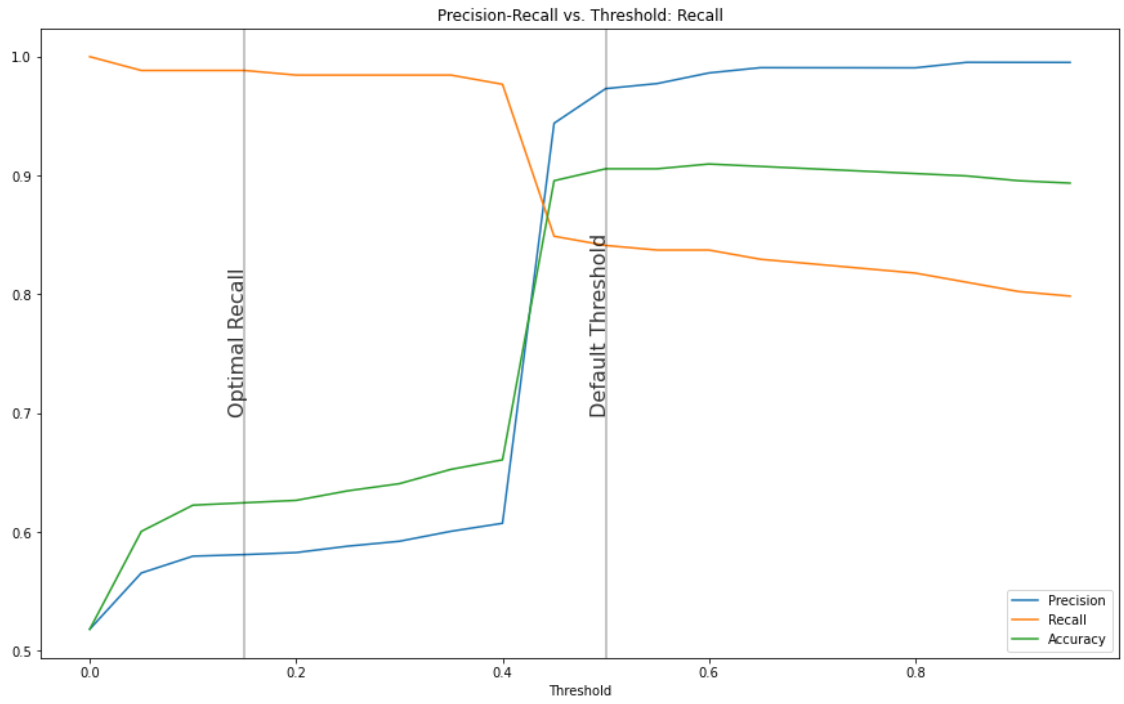
Model	Recall	Precision	Accuracy
Closed/Open with 0.50 threshold	0.98624	0.99537	0.99083
Closed/Open with 0.15 threshold	0.98165	1.0000	0.99083

As you can see this particular model didn't need the threshold to move because it caused the recall to decrease slightly. The precision did increase to 1 even though the focus is on recall.

Below is the confusion matrix of the model at 0.50 and 0.15 threshold. As you can see there weren't any changes.



Below is a precision-recall vs threshold chart showing the default threshold and the optimized threshold with the model metrics



V. Results

For the Open and Closed classes, I would use the Sequential model because the test set performed slightly better than the training set. The MobileNetV2 training set performed slightly better than the testing set. For the yawn and no_yawn classes, I would use the MobileNetV2 model because the test set performed slightly better than the training set. The Sequential training set performed slightly better than the testing set.

Overall both of the models are performing well. The next step is to experiment with the models in an environment that has a higher computational power. With Google Colab if I would add MaxPooling2D or Dropout to the sequential or MobileNetV2 models, it would run out of memory. The next step after that I would transform the model into a real time drowsiness detection that car companies can use to make roads safer.