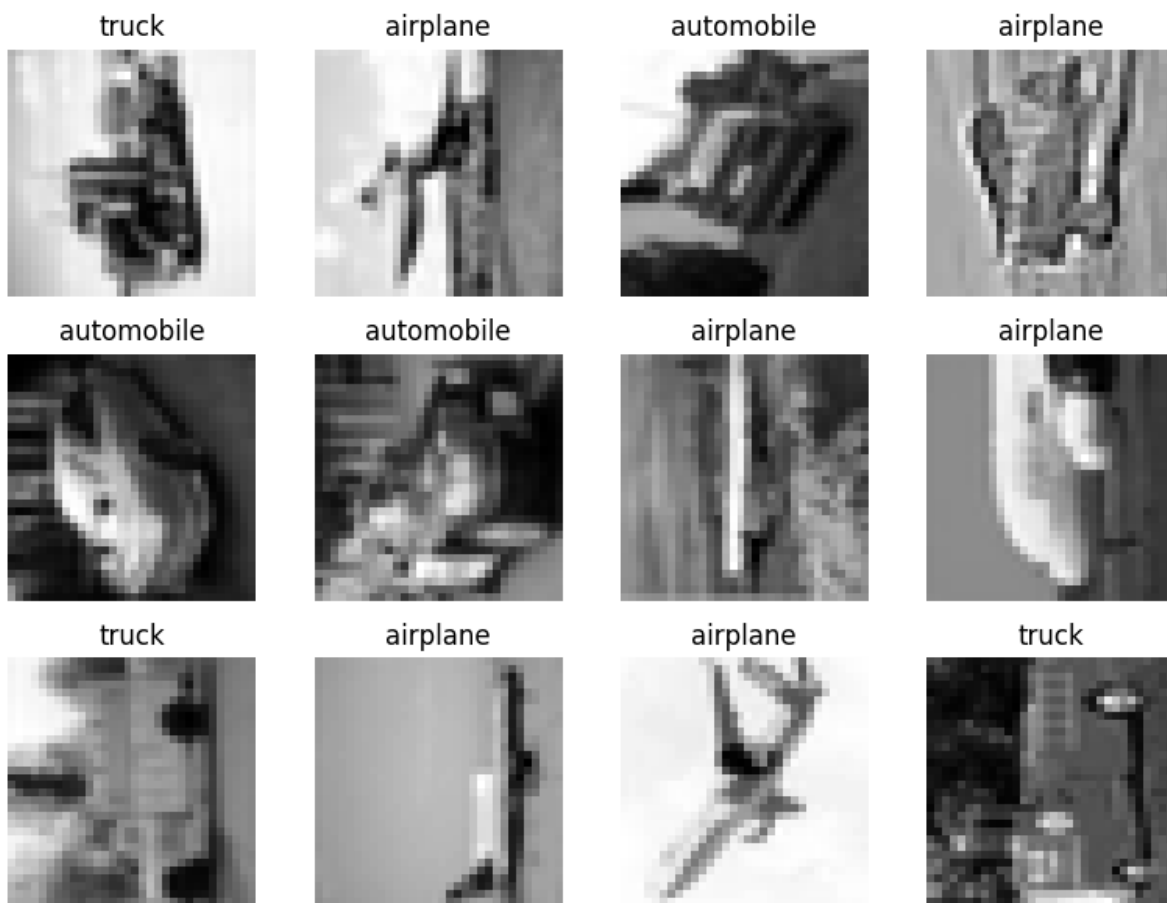


1: Image Recognition using CNN

b.

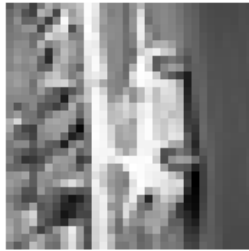


c.

automobile



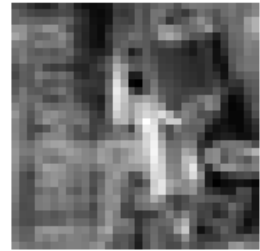
airplane



automobile



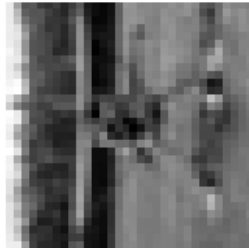
airplane



truck



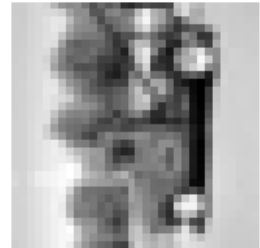
automobile



automobile



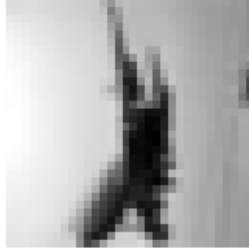
automobile



airplane



automobile



truck



automobile

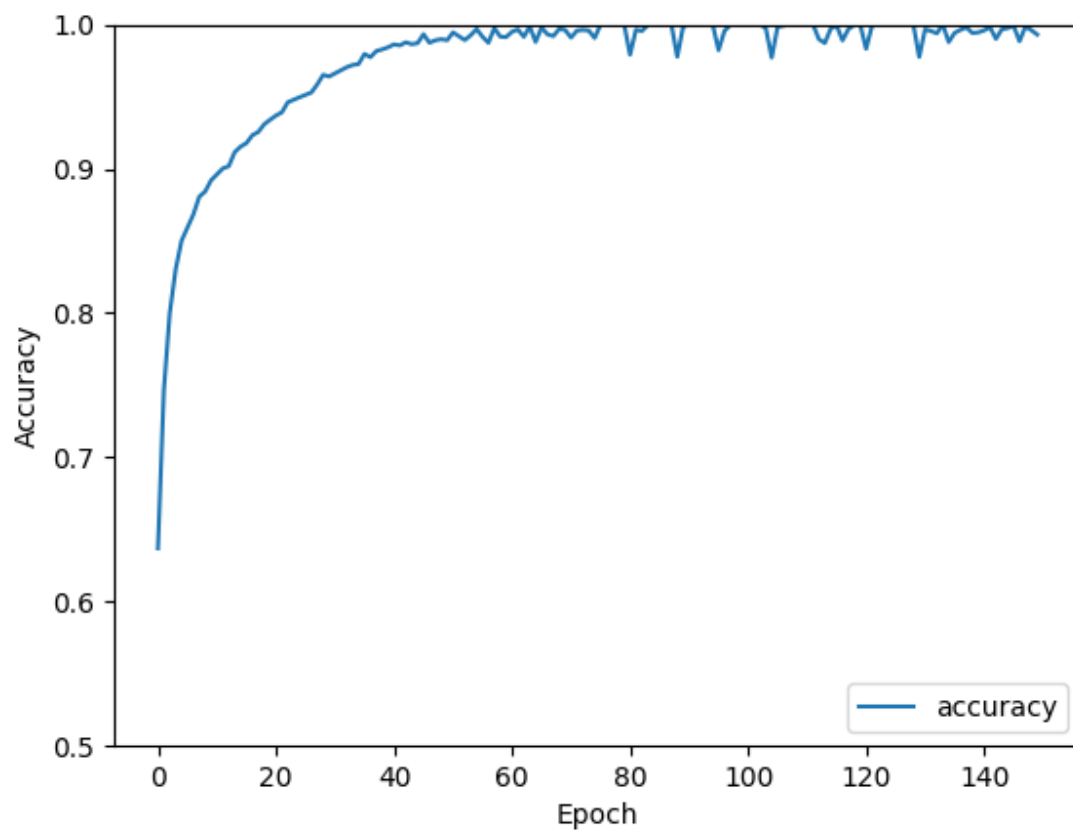


e.

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 38, 38, 32)	320
max_pooling2d (MaxPooling2D)	(None, 15, 15, 32)	0
conv2d_1 (Conv2D)	(None, 11, 11, 32)	25,632
max_pooling2d_1 (MaxPooling2D)	(None, 5, 5, 32)	0
flatten (Flatten)	(None, 800)	0
dense (Dense)	(None, 3)	2,403

Total params: 28,355 (110.76 KB)
Trainable params: 28,355 (110.76 KB)
Non-trainable params: 0 (0.00 B)



accuracy = 0.987

f. accuracy = 0.851

We can increase the testing accuracy by stopping the training earlier so the model does not overfit to the training data.

2. Window Based Recognition:

b.

I used 32 Filters in the first layer at a 5x5 kernel size, and then 10 filters with size 3x3, then a dense network to narrow down to one node.

```
model.add(layers.Conv2D(32, (5, 5), activation='relu', input_shape=imgs[0].shape))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(10, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Flatten())
#model.add(layers.Dense(num_hidden_units, activation = 'sigmoid'))
model.add(layers.Dense(1, activation='sigmoid'))

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 92, 28, 32)	2,432
max_pooling2d (MaxPooling2D)	(None, 46, 14, 32)	0
conv2d_1 (Conv2D)	(None, 44, 12, 10)	2,890
max_pooling2d_1 (MaxPooling2D)	(None, 22, 6, 10)	0
flatten (Flatten)	(None, 1320)	0
dense (Dense)	(None, 1)	1,321

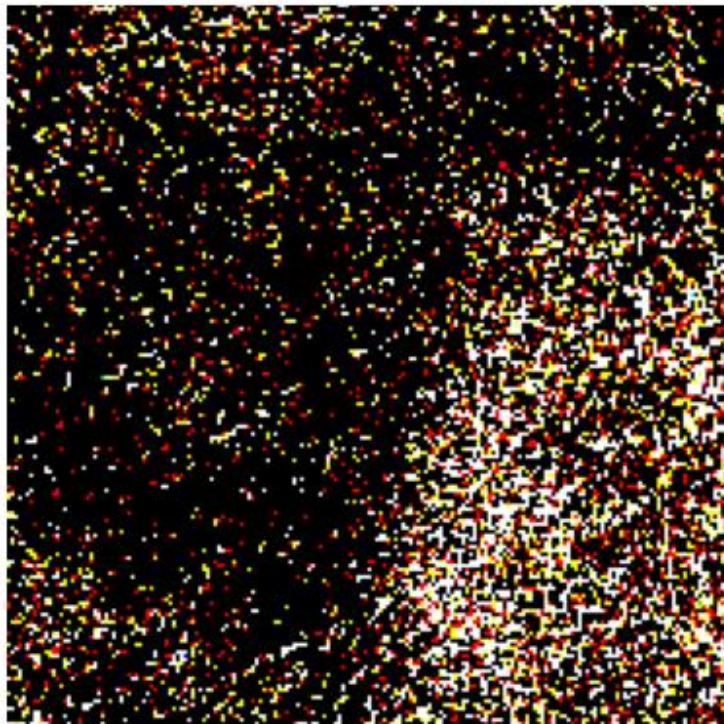
Total params: 6,643 (25.95 KB)
Trainable params: 6,643 (25.95 KB)
Non-trainable params: 0 (0.00 B)

Accuracy = 99.0

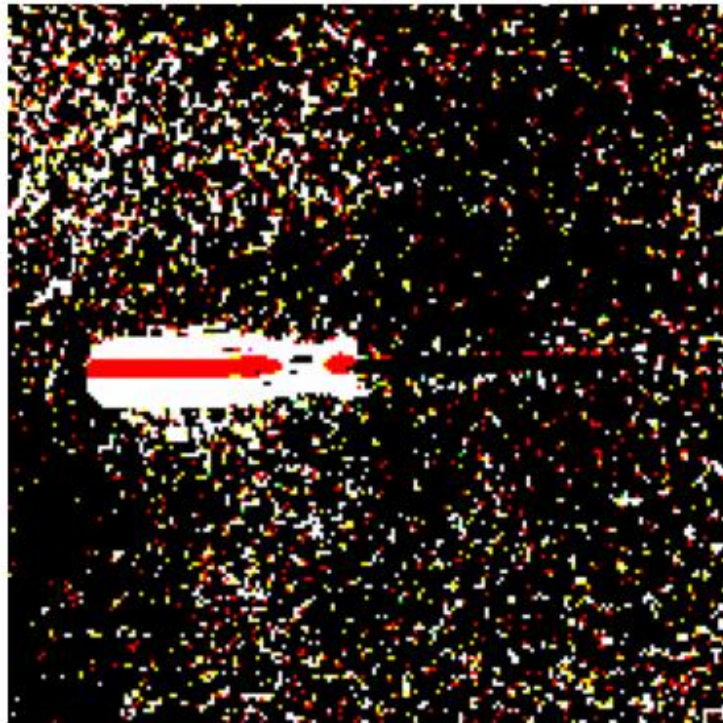
c.

I couldn't figure out why the colors displayed were so off, but cars seemed to be detected

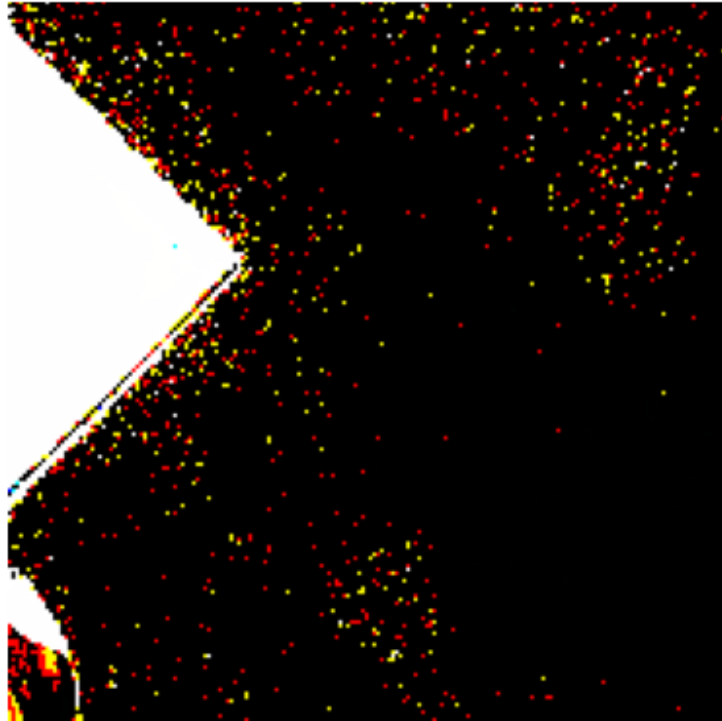
1.



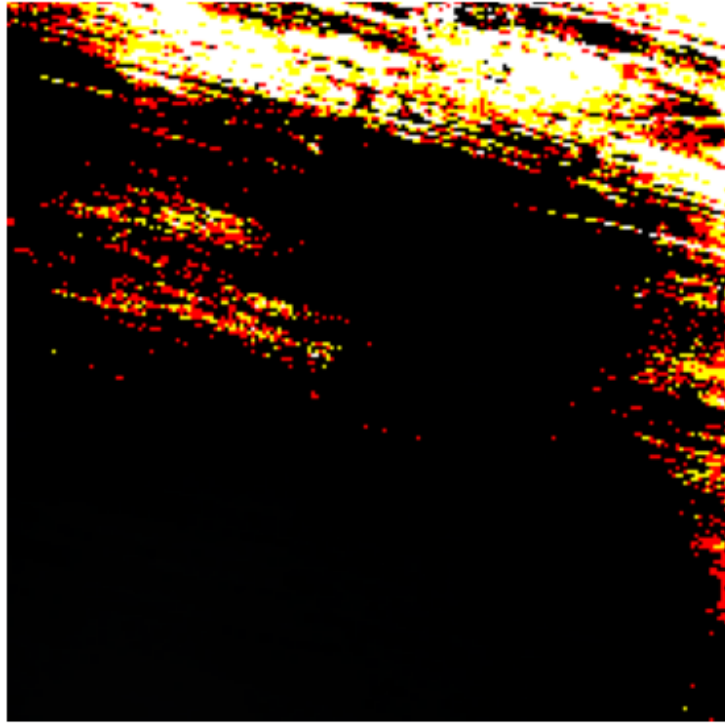
2.



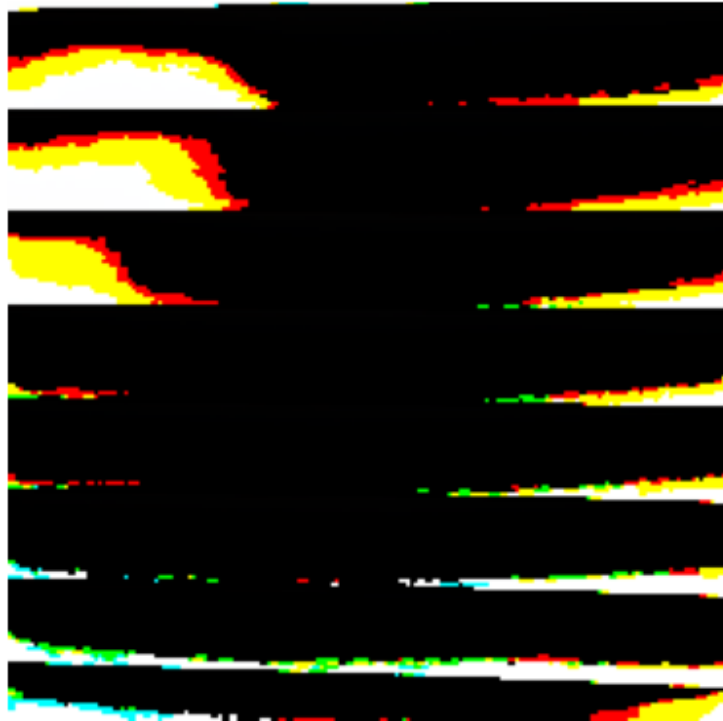
3.



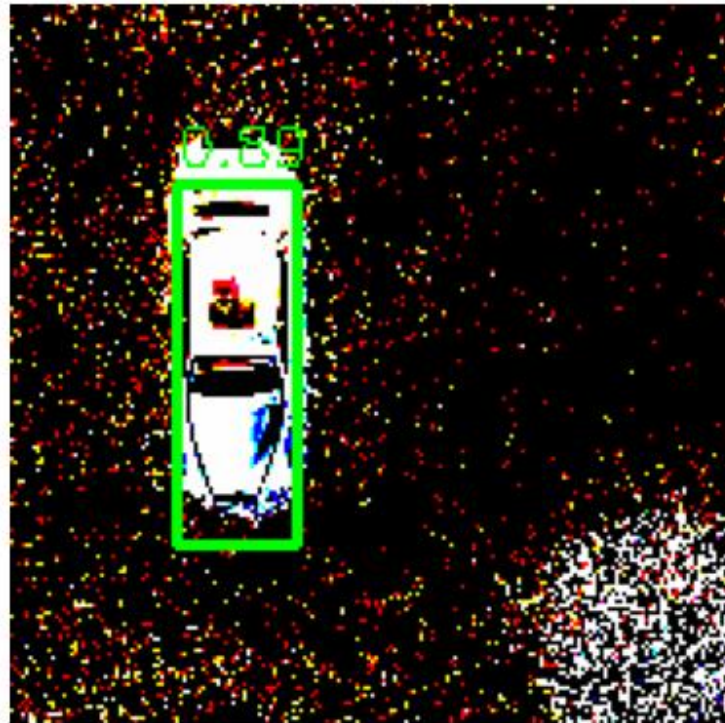
4.



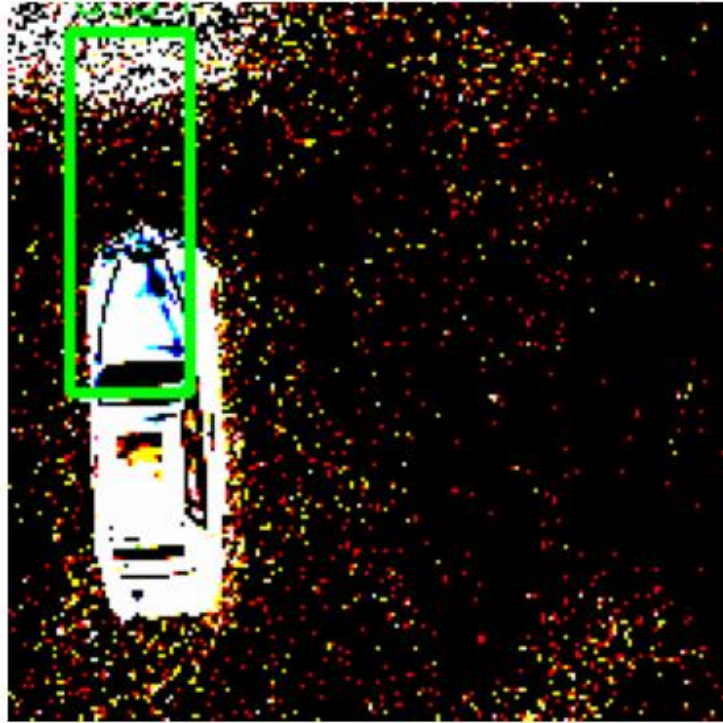
5.



6.



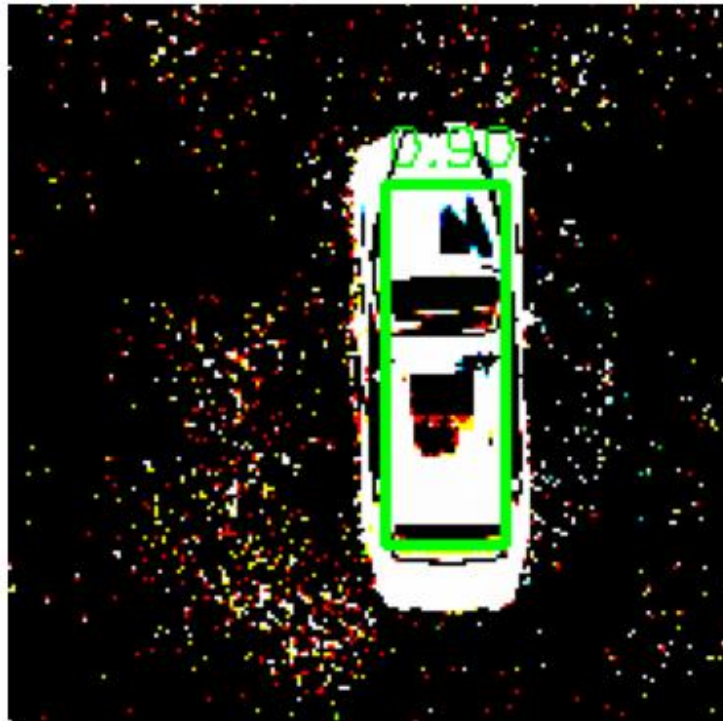
7.



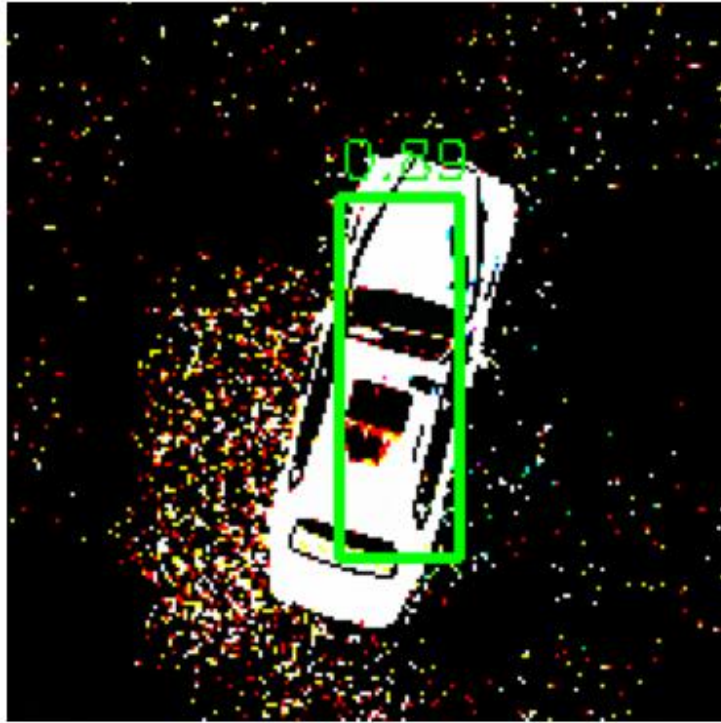
8.



9.



10.



There were no false positives or negatives, but on image 7 the window missed the back half of the car. In order to improve performance, I could add more layers to the model, and stop training earlier so the model is not overfit to the training data.