[No Cap]

Attention Monitoring

Are You Paying Attention?

👀

Tyrone Wilkinson

The Idea

Driver-monitoring cameras will become standard in all autonomous vehicles.

- Tesla vehicles initially only had an “agreement” and hardware embedded in the steering wheel to detect whether pressure was being applied.
- Volvo too was slow to implement.
- They have joined the likes of Comma.ai’s Openpilot, GM’s Super Cruise, Ford’s Co-Pilot360, and a myriad of other companies and their driver-assistance technologies that incorporate driver-monitoring.

The Implementation

- Bring attention monitoring from the car to the home.

The Steps

1. Understand Neural Networks
2. Outline Project
3. Collect and Prepare Data
4. Build, Test, Iterate
5. Final Results

1. Understanding Neural Networks

- Study
- CNN
- Transfer learning
- ResNet-152
- Keras, TensorFlow

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
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

Google Trends

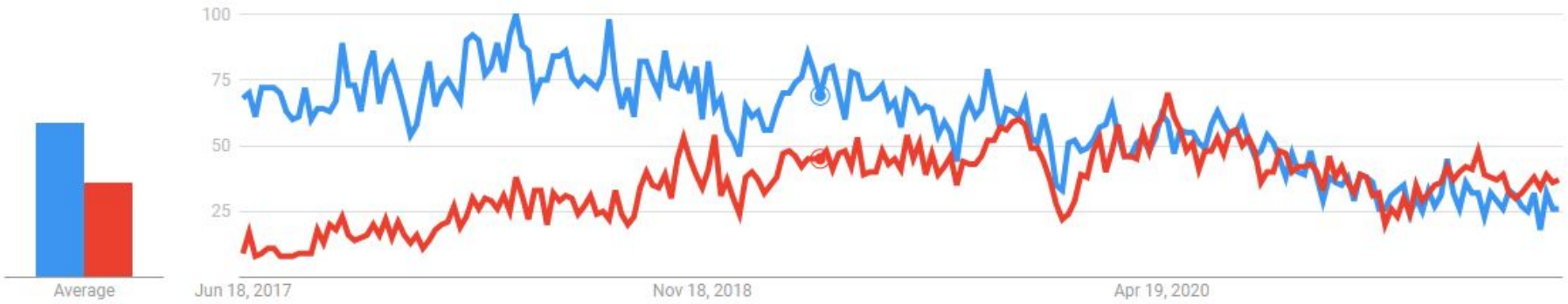
tensorflow
Search term

pytorch
Search term

+ Add comparison

United States | 6/17/17 - 7/17/21 | All categories | Web Search

Interest over time



2. Outline

a. Train a pre-trained CNN to classify images.

b. Use CNN to classify video clips.

c. Use CNN to classify live video.

d. Learn how to train the model on AWS (TBD).

e. Deploy a web application (TBD).

3. THE DATA

https://fei.edu.br/~cet/facedatabase.html
A Brazilian face database that contains a set of face images taken between June 2005 and March 2006 at the Artificial Intelligence Laboratory of FEI in São Bernardo do Campo, São Paulo, Brazil.

http://www.anefian.com/research/face_reco.htm
Georgia Tech face database contains images taken between 06/01/99 and 11/15/99 at the Center for Signal and Image Processing at Georgia Institute of Technology.

https://generated.photos/
An AI-based face generator built on a proprietary dataset of tens of thousands of images of people taken in studio.

https://www.google.com/imghp
Google Images

3. THE DATA

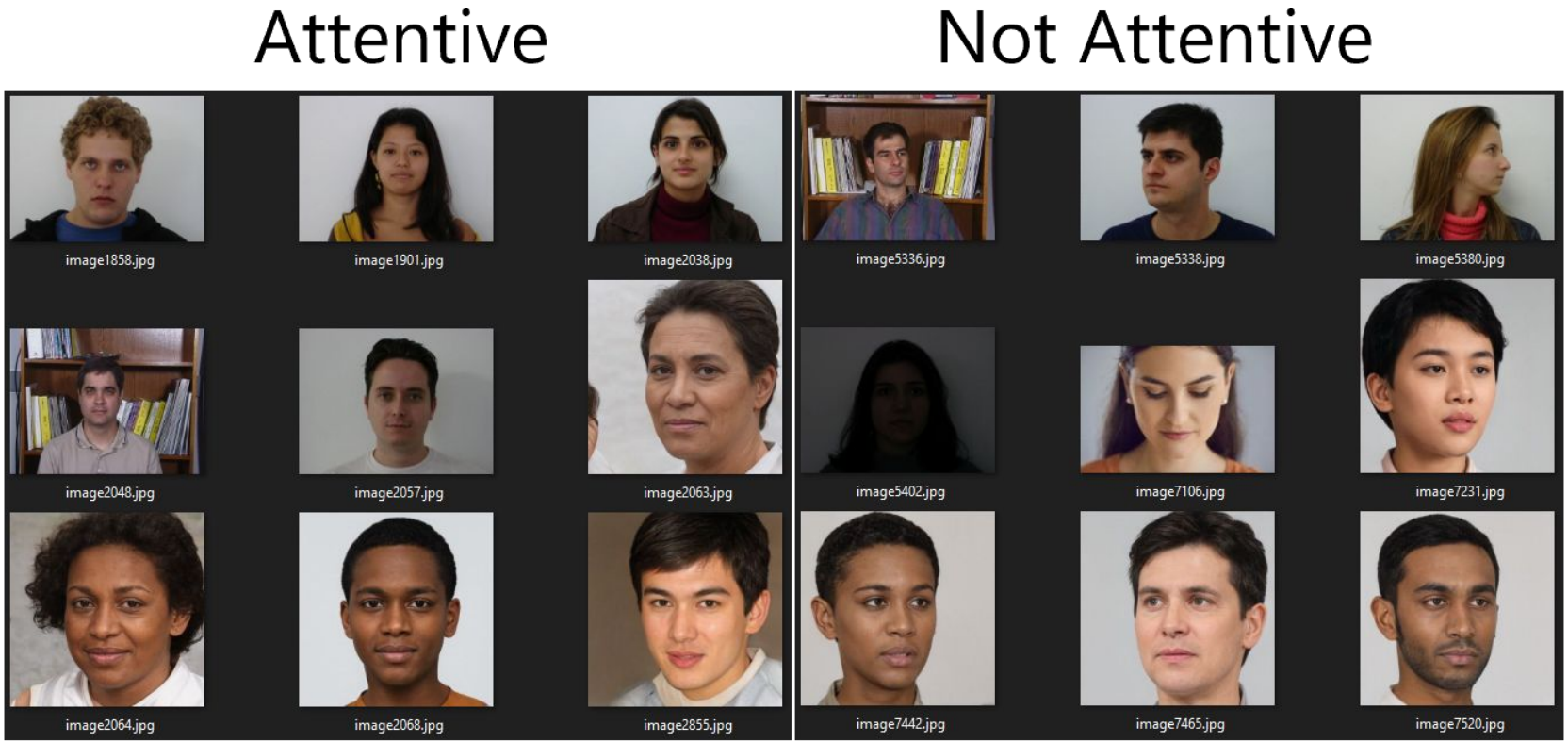
- 10247 photos in total
- 70-15-15 train-validate-test split

Train: 7302 images belonging to 2 classes

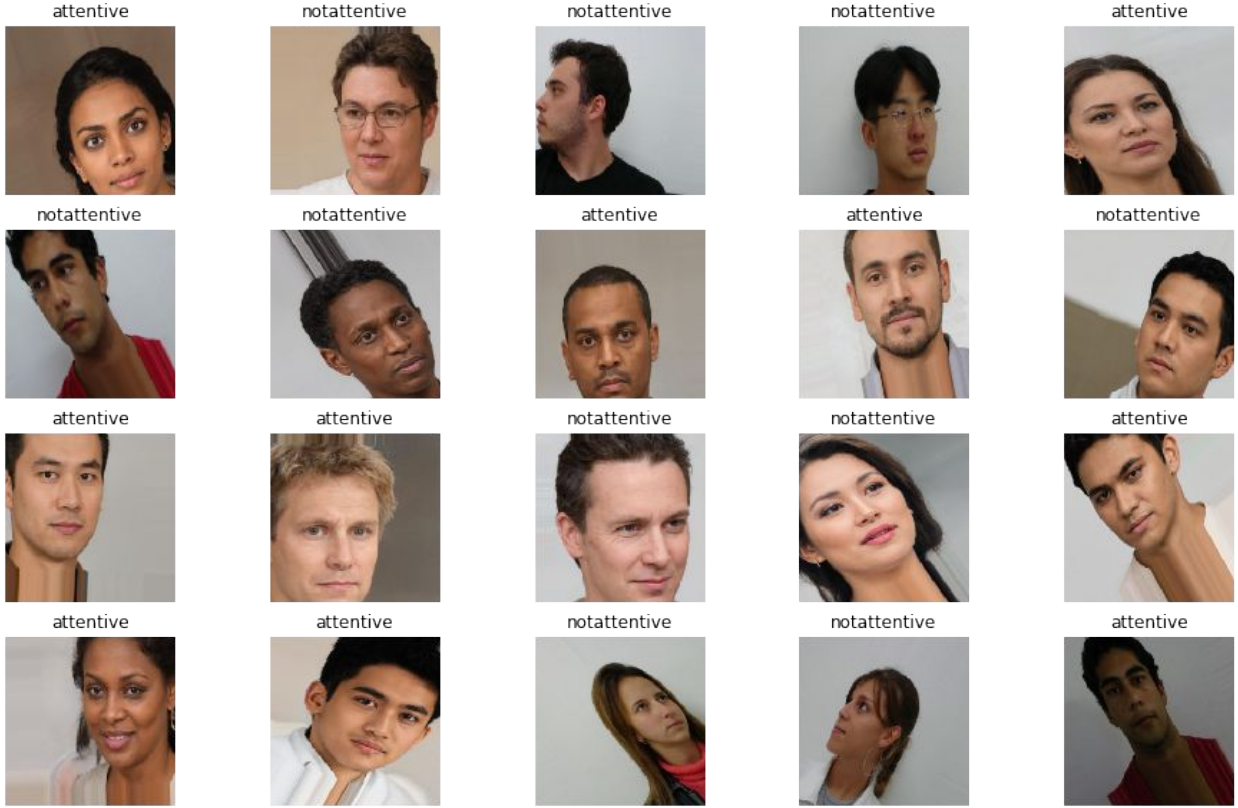
Validate: 1561 images belonging to 2 classes.

Test: 1564 images belonging to 2 classes.

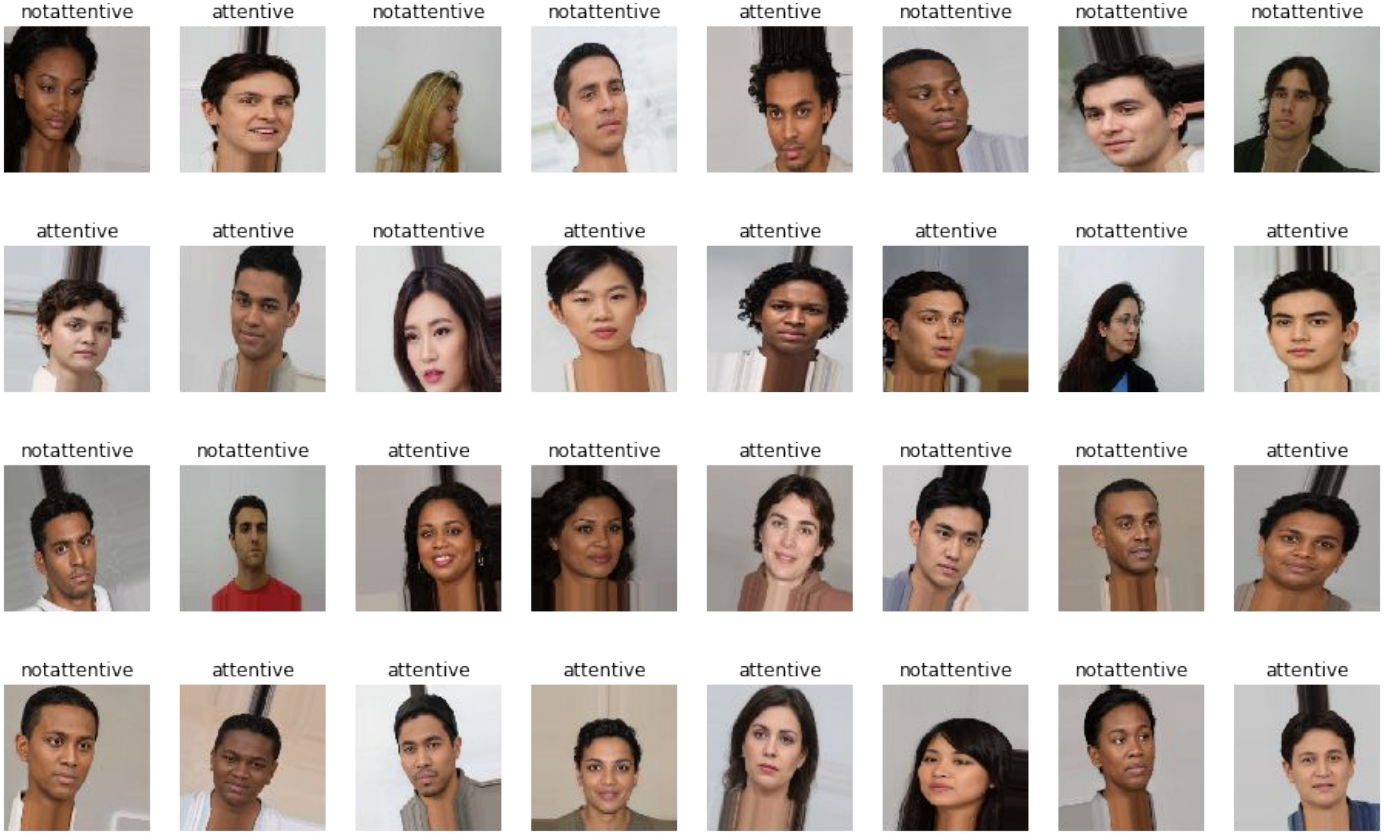
3. THE DATA


Attentive
Not Attentive
image1858.jpg
image1901.jpg
image2038.jpg
image2048.jpg
image2057.jpg
image2063.jpg
image2064.jpg
image2068.jpg
image2855.jpg
image5336.jpg
image5338.jpg
image5380.jpg
image5402.jpg
image7106.jpg
image7231.jpg
image7442.jpg
image7465.jpg
image7520.jpg

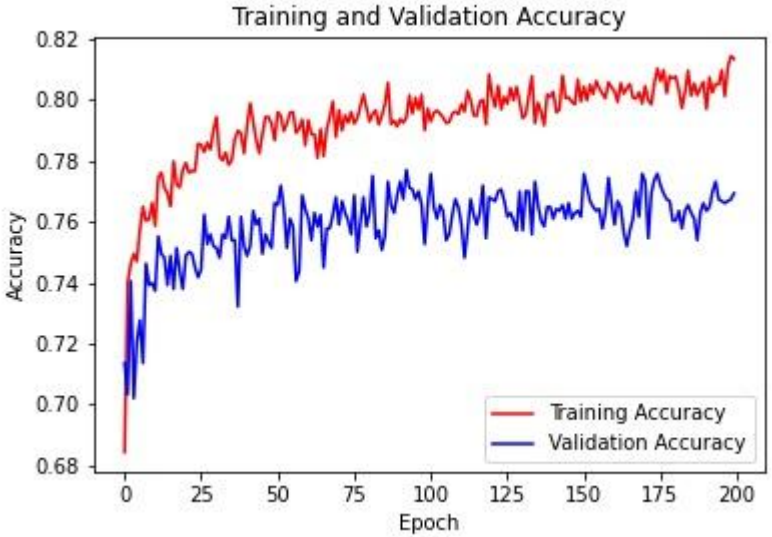
3. THE DATA

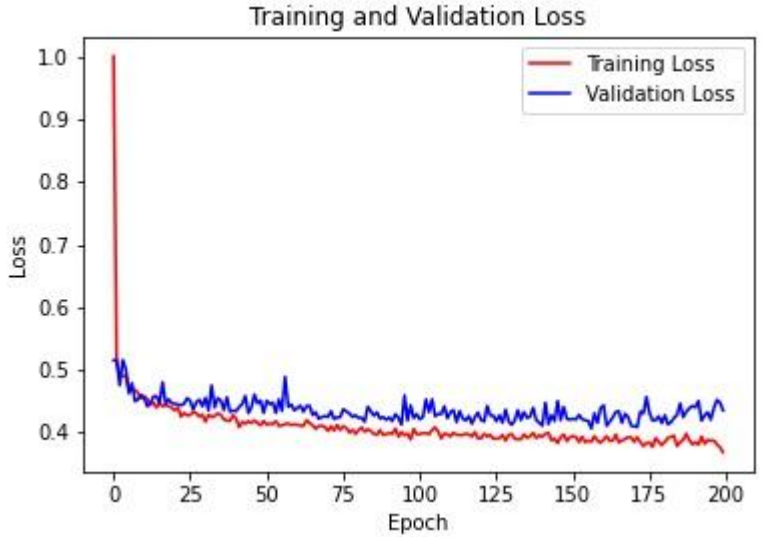

attentive
notattentive
notattentive
notattentive
attentive
notattentive
notattentive
attentive
attentive
notattentive
attentive
attentive
notattentive
notattentive
attentive
attentive
attentive
notattentive
notattentive
attentive

3. THE DATA

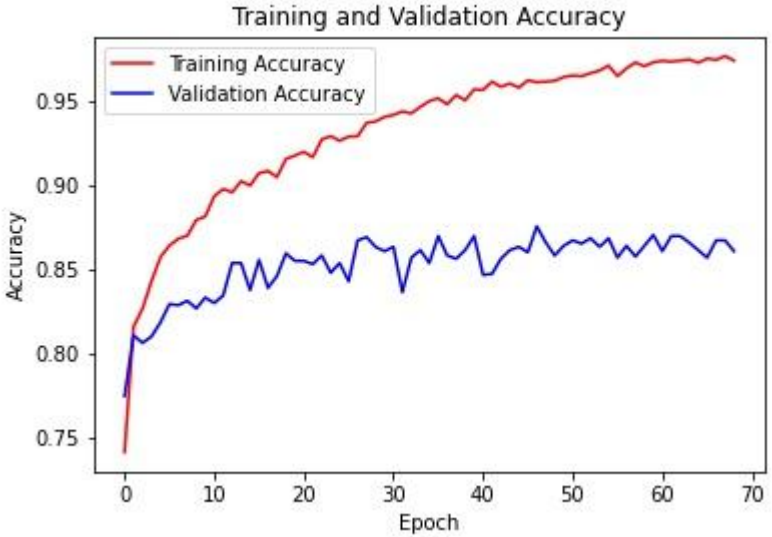

notattentive
attentive
notattentive
notattentive
attentive
notattentive
notattentive
notattentive
attentive
attentive
notattentive
attentive
attentive
attentive
notattentive
attentive
notattentive
notattentive
attentive
notattentive
attentive
notattentive
notattentive
attentive
notattentive
attentive
attentive
attentive
attentive
notattentive
notattentive
attentive

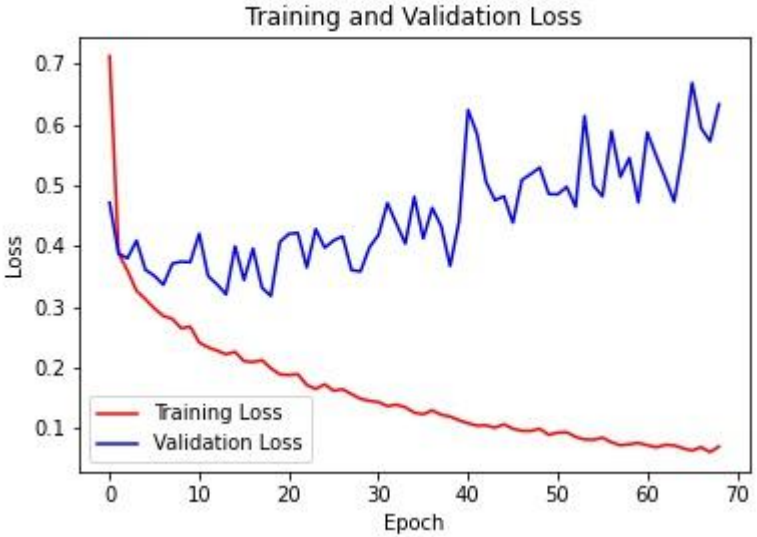
4. Build, Test, Iterate





4. Build, Test, Iterate



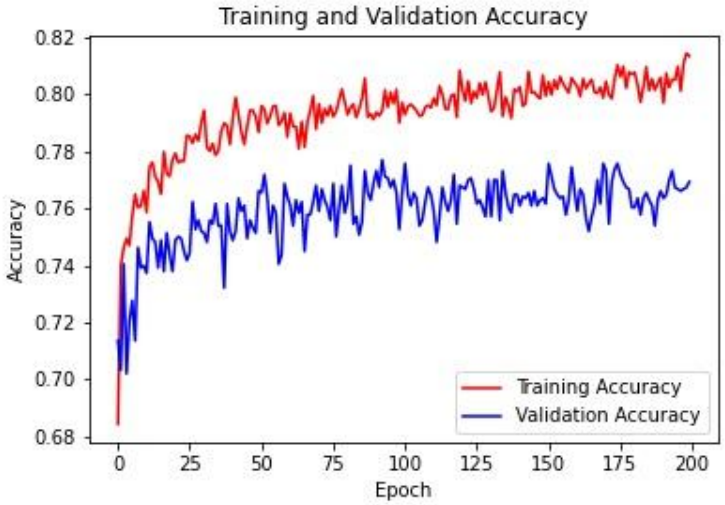


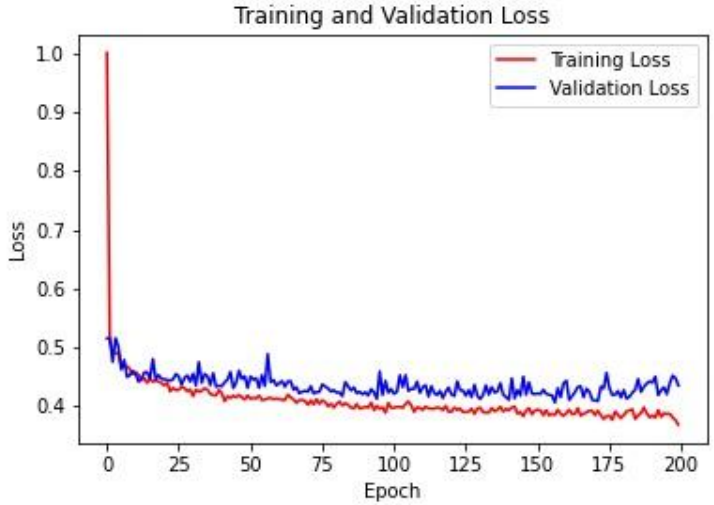
4. Build, Test, Iterate

1st Model

Loss:
0.4209

Accuracy:
0.7615

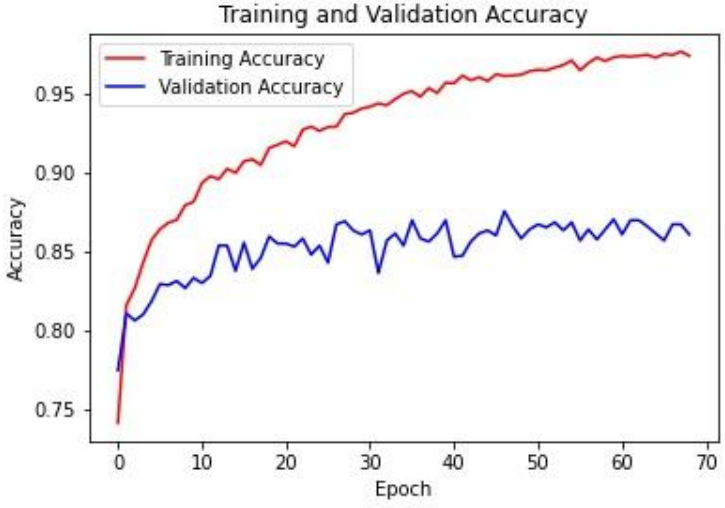


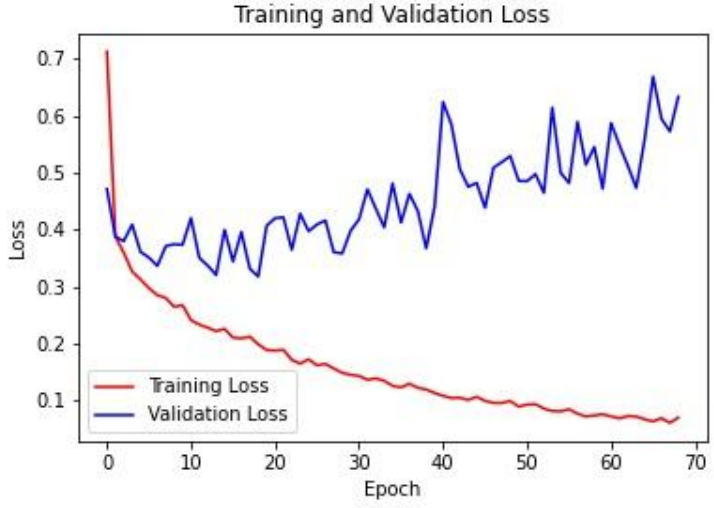


2nd Model

Loss:
0.3175

Accuracy:
0.8561





5. Results

OpenCV computer vision library in Python used for video classification.

Things I noticed while testing:

- “Not Attentive” labeling was better than “Attentive” labeling.
- Distance affected accuracy.
- The location of my pupils did not really affect accuracy.

5. Results

Before change

Note: the prediction values between zero and one correspond to how confident the model is that the image it is classifying deserves a zero label (“ATTENTIVE”) or the one label (“NOT ATTENTIVE”). I multiplied the values by 100--*ignore that.*

5. Results

After change

Note: To test the generalizability of the model, I let my hair down since there were no images in the dataset with my hair type, and put my glasses on since there were few images with people wearing glasses. Neither action seemed to impair the results.

5. Results

What (else) would I have done differently?

- Grayscale vs color?
- Training: AWS > Local
- Better labeling

Late Additions

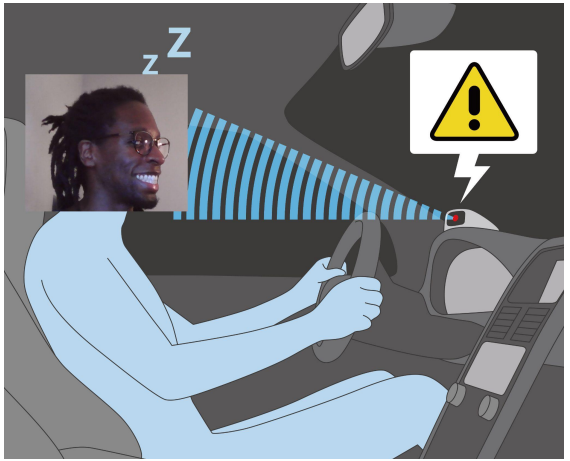
Web App using Streamlit:

[https://share.streamlit.io/tyronewilkinson/attentionmonitoring/webapp/webapp.py]

Training on AWS with PyTorch will roughly entail:

- Create a notebook instance
- Upload the data to an S3 bucket
- Use SageMaker's built-in image classification algorithm that uses ResNet
- Use the ml.p2.xlarge GPU/EC2 instance for training—it only has 1 gpu but that should be sufficient for my dataset
- Learn and use PyTorch

References



Helpful Sources

- NYC Data Science Academy
- StatQuest!!!
- https://www.pyimagesearch.com/
- https://machinelearningmastery.com/
- https://towardsdatascience.com/

Applications Used

- Image Downloader: Batch Image Download Browser Extension
- IrfanView: Batch Image Convert and Rename
- Git LFS: Store Full Dataset on GitHub