ConvHeads

Samyak R Jain
S Sai Chaitanya
Himanshu Kandpal
Aayush Birla

Distracted Driver Detection

AI-2

Problem Statement

- India is the number one contributor to global road crash mortality and morbidity figures.
- Every hour, 16 lives are lost to road crashes in India.
- According to the CDC, 1 in 5 car accidents is caused by a distracted driver in the US.
- Sadly, this translates to 425,000 people injured and 3,000 people killed by distracted driving every year.

47%

People use phones while driving

20%

People have had a near-miss or a crash while using a phone while driving

Source: Distracted Driving in India Survey [1]

Proposed Solution

- A system that **alerts the driver** if they are distracted.
- The system **should be accurate enough** to be practically useful.

Texting on a phone

- We train and use **Convolutional Neural Networks** to detect whether a driver is distracted or not.
- We also use CNNs to predict how the driver is distracted - **texting, talking on phone, drinking**, etc.
- We use data from State Farm Distracted Driver Dataset on Kaggle.

Dataset

- We used the State Farm Distracted Driver Detection Dataset from Kaggle.
- This dataset has 480x640 **images well distributed among 10 classes.**
- The images were collected in a controlled environment - a truck dragging the car around on the streets - so these "drivers" weren't really driving.
- Train Set - 17950 images (80%)
- Validation Set - 4483 images (20%)

Sample Train Images with labels

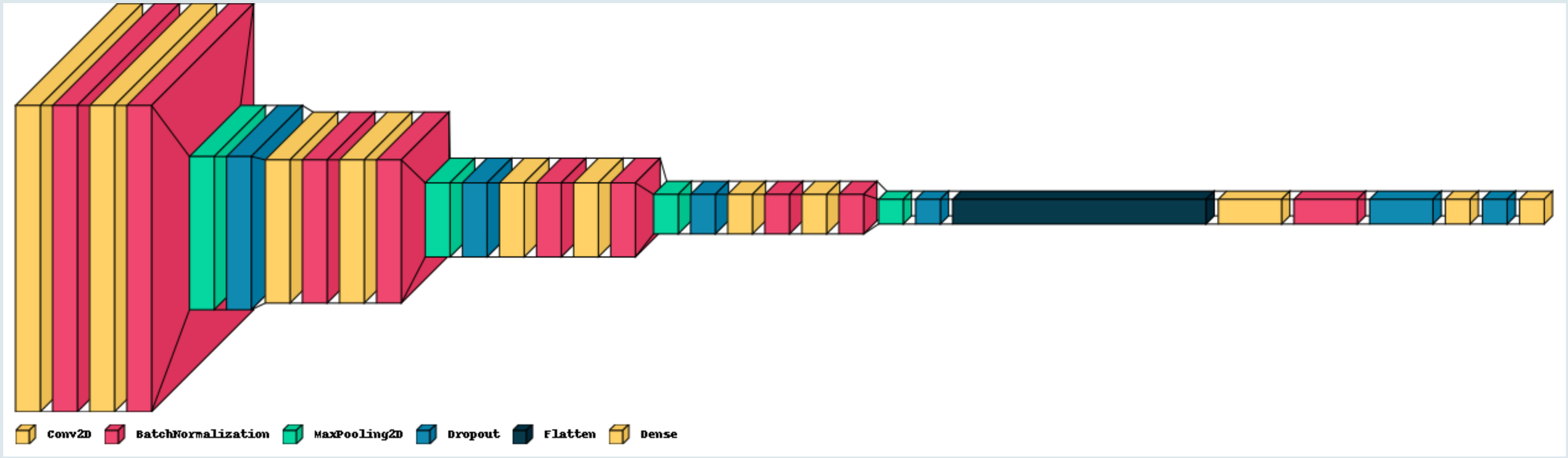
Safe Driving

Drinking

Sample Real World Test Images
(collected by us)

Operating the radio

A CNN model from Scratch

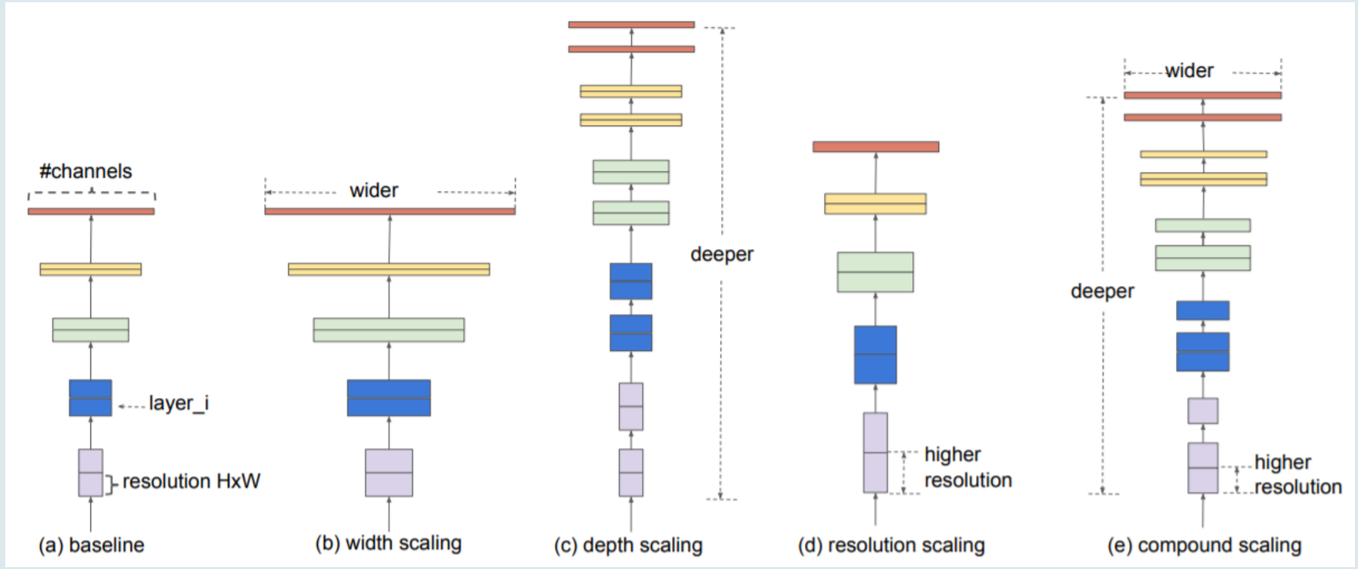


Model Architecture

- Model Parameters: 1.4M
- 4 blocks of:
  - CONV-BN-CONV-BN-MAXPOOL-DROPOUT
- Number of filters used for each block: 16, 32, 64, 128
- Convolutional Filter size used throughout - (3x3)
- 8 CONV Layers
- 3 DENSE Layers

- We reduced the Image Size from (640x480) to (64x64) to reduce the dimensionality and faster training of the model
- Batch Size = 32
- Optimizer used : Adam
- Learning Rate : 0.001
- We trained the model for 15 Epochs
- Training time: 51 mins

Fine-Tuned EfficientNetB3 Model



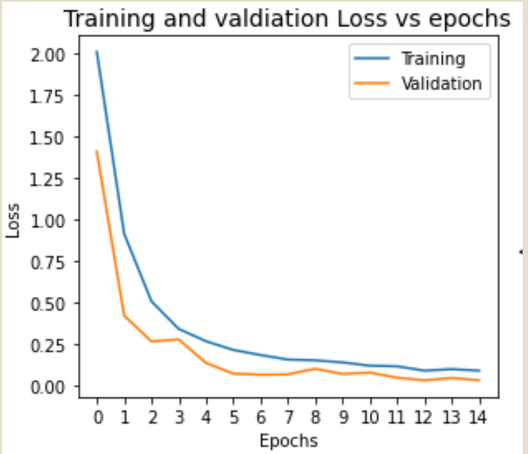
- We have experimented with multiple pre-trained models and EfficientNetB3 gave the best results. (**10M Parameters**)

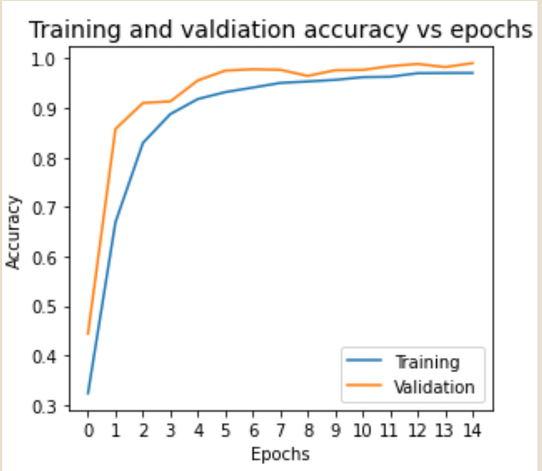
  EfficientNets provide a solution on how to scale up Convolutional neural networks, they use a compound scaling method that **uniformly scales depth, width and resolution.**
- We have fine-tuned all the layers of the model with a lower learning rate 1e-4.
- We have chosen EfficientNetB3 in the EfficientNets family because it's recommended input shape 300 is most compatible with the scaled down input shape 240x320 of the images.
- We have achieved a training accuracy of 98.97% and a validation accuracy of 98.97%.
- The model has trained for 2 hour 40 mins. [10 epochs]

Results

Metric	CNN Model from Scratch	Fine-tuned EfficientNetB3
Train Accuracy	97.09%	98.97%
Validation Accuracy	**99.06%**	**98.97%**
Train Loss	0.0936	0.0333
Validation Loss	0.0357	0.0340

Training History - CNN Model from Scratch





- From the above plots, it is clear that the training and validation loss & accuracy converge well and hence the model is **NOT overfitting/underfitting**.

Results

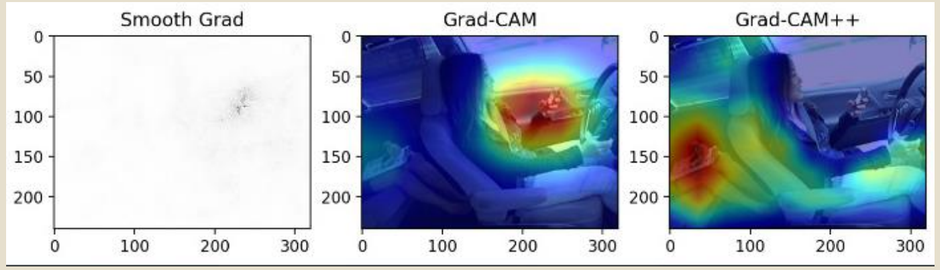
Predictions on real world data collected by us

Observations:

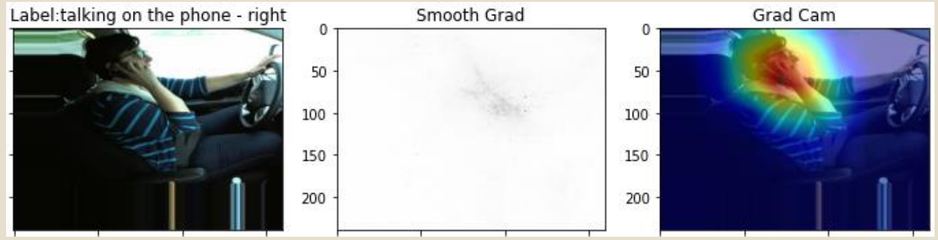
- The model predicted 3 of the above 5 images correctly
- For the misclassified images: the glass in drinking class in train data consists of a single white colored starbucks-like cups.
- Our model could not detect the images with orange glass as drinking. Instead, it has predicted it as a texting on a phone.

Visualizing the predictions

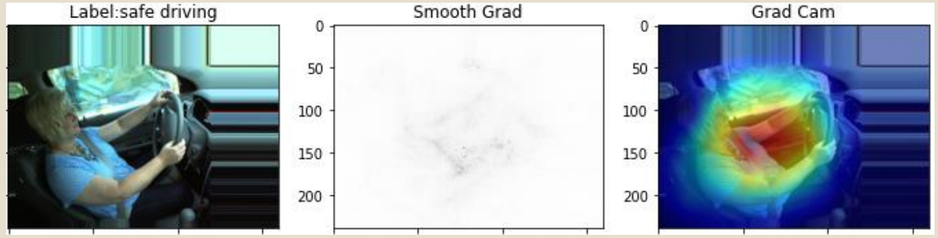
Correctly Classified Images



- We see that the model predicts "texting - left" correctly based on detecting 2 phones (1 in driver's hand and the other phone with the rider at the back)

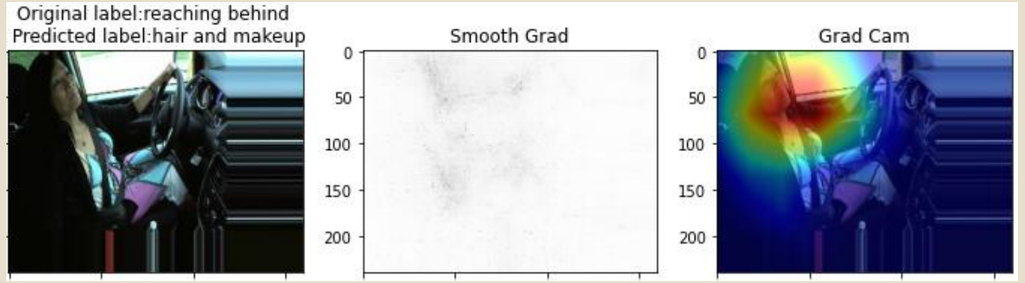


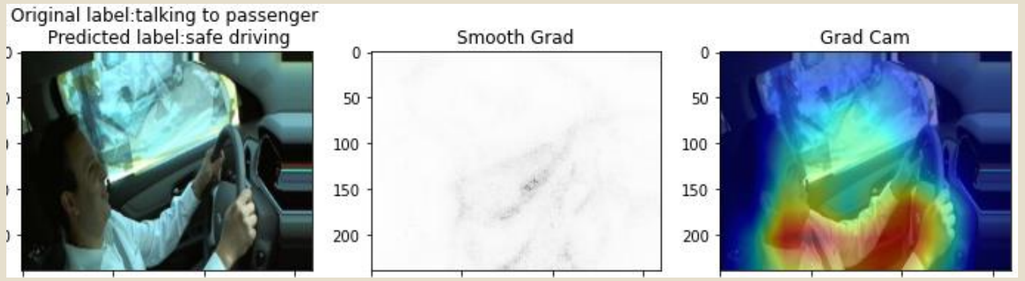
- The model seems to be looking at the right place to make the correct prediction of "talking on phone - right". It is looking at the hand in the phone near the ear.

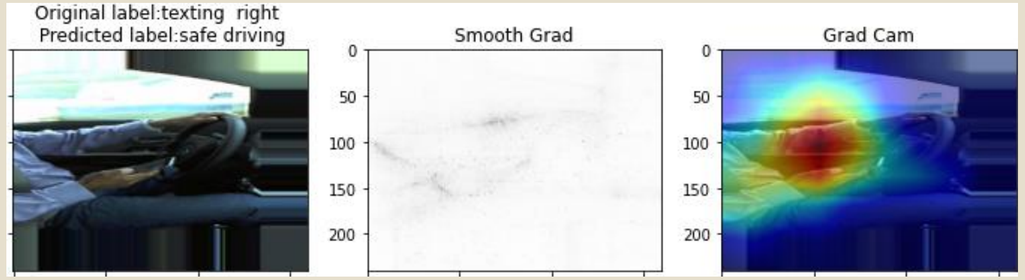


- The model is looking at both the hands being on the steering wheel to correctly predict that the driver is driving safely.

Misclassified Images







Observations:

- In most of above incorrectly predicted cases, the model is looking at the wrong part of the image to make the predictions.
- This could be because the model could not differentiate some tiny differences in the image.
- Also, the augmentations we performed on the data have taken the most important part of the image to make the prediction out of the frame.

Conclusion & Future Work

- We used Convolutional Neural Networks to classify the driver's actions using images from in-car cameras.
- The model has classified images collected from the same environment as the training images, almost perfectly.
- But for a considerable number of images collected from real world, the model has classified the images incorrectly.
- This could be because the data-set only contains images collected from less number of cars, from a specific angle and a few number of drivers.
- If we can train our model with more diverse data like from drivers with a wide range of age, race and physical body types, our model will classify real world images accurately.
- A real world application could be built and deployed in cars based on the model to alert the drivers when distracted.

References

01	Distracted Driving in India - A Study on Mobile Phone Usage, Patterns & Behaviour
02	State Farm Distracted Driver Detection - Kaggle
03	EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks

Technologies and Libraries used

	Tensorflow, Keras, tf-keras-vis
	visualkeras, Numpy, Pandas, Matplotlib and Pillow