

Distracted driver detection

CSC 698-08: Capstone Project

Under the Esteemed Guidance of

Prof. Qianqian Tong

Submitted By

Deepika Dasaroju

888078129



Table of Contents

[Abstract: 1](#_Toc162531980)

[Introduction: 1](#_Toc162531981)

[Data Exploration: 2](#_Toc162531982)

[Class Distribution: 2](#_Toc162531983)

[Image Characteristics: 3](#_Toc162531984)

[Dataset Partitioning: 3](#_Toc162531985)

[Modeling: 3](#_Toc162531986)

[Base Model Greyscale: 4](#_Toc162531987)

[Base Model Greyscale with Early Stopping: 6](#_Toc162531988)

[Base Model RGB: 7](#_Toc162531989)

[RGB Model with Batch Normalization: 9](#_Toc162531990)

[ResNet152: 11](#_Toc162531991)

[VGG16: 13](#_Toc162531992)

[Results: 15](#_Toc162531993)

[Future Scope: 21](#_Toc162531994)

[Final Remarks: 21](#_Toc162531995)

[Conclusions: 21](#_Toc162531996)

[References: 21](#_Toc162531997)

List of Figures

[Figure 1: Sample Image matrix for each class 4](#_Toc162467638)

[Figure 2: Training Metrics plot for Base Model using greyscale images 7](#_Toc162467639)

[Figure 3: Training Metrics plot for Base Model using greyscale images and early stopping 8](#_Toc162467640)

[Figure 4: Training Metrics plot for Base Model using color (RGB) images 10](#_Toc162467641)

[Figure 5: Training Metrics plot for Base Model using color (RGB) images and batch normalization 12](#_Toc162467642)

[Figure 6: Training Metrics plot for ResNet152 model 13](#_Toc162467643)

[Figure 7: Training Metrics plot for VGG16 model 15](#_Toc162467644)

[Figure 8: Model wise Accuracies plot for Train, Validate and Test datasets 16](#_Toc162467645)

[Figure 9 : Model wise Test Accuracies plot 17](#_Toc162467646)

[Figure 10 : Model-wise Loss plot for Train, Validate and Test datasets 18](#_Toc162467647)

[Figure 11: Model-wise Average F1 Score 20](#_Toc162467648)

[Figure 12 : Model-wise false Safe driving prediction rate 21](#_Toc162467649)

List of Tables

[Table 1: Model wise Accuracies 17](#_Toc162467721)

[Table 2 : Model wise Losses 18](#_Toc162467722)

[Table 3 : Model wise Average F1 Values 20](#_Toc162467723)

[Table 4 : Model-wise false Safe driving prediction rate 21](#_Toc162467724)

# Abstract:

The "State Farm Distracted Driver Detection" project utilizes deep learning architectures and techniques to identify and classify driver behaviors that could indicate distraction. By analyzing a dataset of driver images, the project seeks to develop a model capable of detecting various forms of distractions, such as texting, eating, or talking to passengers, with high accuracy. Utilizing a dataset provided by State Farm, the project evaluates models, including grayscale and RGB base models, both with and without early stopping, alongside advanced architectures like ResNet152 and VGG16. Additionally, normalization techniques are explored for their role in optimizing input data representation, aiming to enhance model training efficiency and accuracy. The study aims to identify the most effective methods for classifying driver distractions, leveraging normalization techniques to enhance model training and performance. This report outlines the process from data preprocessing to model evaluation, detailing the findings and their implications for improving road safety.

# Introduction:

Distracted driving has become a significant public safety concern in the constant connectivity and multitasking era. The National Highway Traffic Safety Administration (NHTSA) reports that thousands of fatalities annually on U.S. roads can be attributed to distracted driving, highlighting an urgent need for effective detection and prevention solutions. Against this backdrop, the "State Farm Distracted Driver Detection" project seeks to leverage cutting-edge machine learning techniques to identify and classify behaviors indicative of driver distraction accurately.

This project is motivated by the potential to significantly reduce the incidence of distracted driving accidents through automated detection systems. By accurately identifying distracted behaviors from visual data, such systems can enable real-time interventions, such as alerts to drivers or fleet managers, thereby enhancing road safety. The project harnesses a diverse dataset provided by State Farm, consisting of driver images captured in various states of attention and distraction. This dataset forms the foundation for developing and evaluating a range of deep-learning models.

Central to our approach is the exploration of multiple model architectures and image processing techniques. This includes the development of base models for greyscale and RGB images, which serve as benchmarks for evaluating the impact of image color depth on model performance. Additionally, the project employs early stopping mechanisms to mitigate overfitting, ensuring that our models generalize well to new, unseen data. Normalization techniques are also applied to optimize the input data, enhancing the efficiency and effectiveness of model training.

Moreover, the project evaluates the performance of advanced convolutional neural network (CNN) architectures, namely ResNet152 and VGG16, renowned for their deep learning capabilities and success in complex image classification tasks. These models are chosen for their potential to capture nuanced features and patterns in the data that may indicate distracted driving behaviors.

This introduction sets the stage for the detailed exploration and analysis that follows, outlining the project's objectives, methodologies, and the anticipated impact of its findings on the field of road safety and beyond. Through this work, we aim to contribute to developing intelligent systems capable of reducing the prevalence of distracted driving and enhancing the safety of roads for all users.

# Data Exploration:

The foundation of the "State Farm Distracted Driver Detection" project is a comprehensive dataset provided by State Farm, featuring images of drivers in various states of attention and distraction. This dataset comprises 22,424 images, each categorized into one of ten classes corresponding to specific driver activities and each representing a different driving behavior, such as safe driving, texting, talking on the phone, eating, and other activities that might distract a driver.

## Class Distribution:

* c0 (safe driving): 2,489 images
* c1 (texting - right): 2,267 images
* c2 (talking on the phone - right): 2,317 images.
* c3 (texting - left): 2,346 images.
* c4 (talking on the phone - left): 2,326 images.
* c5 (operating the radio): 2,312 images.
* c6 (drinking): 2,325 images
* c7 (reaching behind): 2,002 images.
* c8 (hair and makeup): 1,911 images
* c9 (talking to passenger): 2,129 images.

The distribution of images across the classes shows a relatively balanced dataset, which is crucial for training unbiased models. However, slight imbalances, such as fewer images in classes c7 ('reaching behind') and c8 ('hair and makeup'), might require attention during model training to ensure fair representation and learning.

Figure 1. shows one sample image for each class.

A collage of people driving

Description automatically generated

Figure 1: Sample Image matrix for each class

## Image Characteristics:

Each image in the dataset is initially in RGB format with a resolution of 480x640 pixels. For model training and evaluation purposes, images are resized to 120x160 pixels to standardize input dimensions and reduce computational load. Two image formats are explored:

* **Greyscale Images:** Converted from RGB to greyscale to reduce complexity and focus on textural and shape features. The greyscale images have a single channel reflected in their shape (120, 160).
* **RGB Images:** Maintained in their original color format, providing rich color information that could be beneficial for distinguishing certain distracted behaviors. These images have three channels (RGB) with a (120, 160, 3) shape.

## Dataset Partitioning:

To effectively train, validate, and test the developed models, the dataset is partitioned into three subsets:

* **Training Set:** This consists of 14,351 samples utilized for training the models.
* **Validation Set:** Comprises 3,588 samples, used to tune hyperparameters and monitor model performance to avoid overfitting.
* **Test Set:** This set contains 4,485 samples, serving as an unbiased resource for evaluating the models' generalization capabilities on unseen data.

This partitioning ensures a robust framework for model development, allowing for iterative refinement of models based on performance metrics evaluated on the validation set and final performance assessment on the test set.

# Modeling:

This section describes the machine learning models explored, focusing on the architecture that yielded the best performance. The report delves into the specifics of Convolutional Neural Networks (CNNs), highlighting the architecture's layers, activation functions, and the rationale behind certain hyperparameters. It also covers other models and techniques tested, providing a comparative analysis to underscore the effectiveness of the selected model. Key aspects such as the training process, loss functions, optimization algorithms, and strategies to combat overfitting, such as dropout and regularization, are highlighted.

## Base Model Greyscale:

The "Base Model Greyscale" is a convolutional neural network (CNN) designed to identify distracted driver behaviors from greyscale images. Given the reduced complexity of greyscale images compared to their RGB counterparts, this model focuses on extracting relevant features without the additional computational overhead associated with color processing.

**Input:**

The model takes an input shape of (120,160,1), corresponding to the resized greyscale images, which are 120 pixels in height and 160 pixels in width and have a single-color channel (greyscale).

**Architecture:**

* **Convolutional Layers:**
  + The first convolutional layer has 128 filters with a kernel size of (3,3) and utilizes the 'relu' activation function. It is designed to capture high-level features from the input image.
  + This is followed by a max pooling layer with a pool size of (2,2), which reduces the spatial dimensions by half to (59×79).
  + The second convolutional layer consists of 64 filters with a (3,3) kernel and 'relu' activation, aiming to refine the feature maps further.
  + Another max pooling layer follows, again halving the spatial dimensions to 28×38.
  + The third convolutional layer has 32 filters, a (3,3) kernel, and 'relu' activation, allowing the network to detect even more nuanced features.
  + The final max pooling layer brings down the spatial dimension to 13×18.
* **Flatten Layer:**
  + The output of the last pooling layer is flattened into a 1D vector to serve as input for the subsequent dense layers. The flattened layer has 7488 units.
* **Dense Layers:**
  + The first dense layer has 512 units with 'relu' activation, serving as a fully connected layer that can learn non-linear combinations of the high-level features identified by the convolutional layers.
  + The second dense layer and the output layer have 10 units corresponding to the 10 classes of driver behaviors, with 'softmax' activation to output the probability distribution over the classes.

**Output:**

The final layer outputs a 10-dimensional vector representing the probabilities of the input image belonging to each of the 10 classes.

**Trainable Parameters:**

The model has 3,933,034 trainable parameters, which indicates the complexity and learning capacity of the network.

**Training Procedure:**

The model training utilizes the 'adam' optimizer and the 'sparse\_categorical\_crossentropy' loss function, with 'accuracy' as the evaluation metric. Image data is normalized during training by scaling pixel values to the range [0, 1] for improved training dynamics.

**Model Evaluation:**

The evaluation of the "Base Model Greyscale" on the test set provides the following results:

* **Loss**: The model has a significantly high-test loss of 320.1765. This value is substantially larger than typical loss values and suggests that there might have been an issue during the evaluation process or with how the loss was recorded or scaled.
* **Accuracy**: The model achieved a test accuracy of 98.55%, which is a remarkable result and indicates that the model was highly effective in correctly classifying the various distracted driving behaviors in the greyscale image test set.

Figure 2 shows how loss and accuracy metrics for training and validation datasets vary throughout the training.

From the loss curves, it is noticed that the validation loss diverged from the training loss value starting from 15th epoch. This usually occurs due to overfitting, and given the discrepancy between the high accuracy and the unusually high loss value, it would be essential to investigate further. The high loss could indicate anomalies in specific predictions that contribute excessively to the loss metric or an overfitting issue where the model is not generalizing well despite the high accuracy. It may also point to a potential error in the loss calculation or data handling during testing.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated

Figure 2: Training Metrics plot for Base Model using greyscale images

## Base Model Greyscale with Early Stopping:

The "Base Model Greyscale with Early Stopping" is a convolutional neural network (CNN) designed to detect distracted driver behaviors from greyscale images. This model incorporates an early stopping mechanism during training to prevent overfitting, which enhances the model's generalization ability to new, unseen data.

This model uses the same input architecture, including the outputs and number of trainable parameters. The only enhancement is to train with the Early stopping callback.

**Early Stopping Mechanism:**

The early stopping callback monitors the validation loss, halting the training process if there is no improvement in a specified number of epochs (patience parameter). In this case, the patience is set to 3 epochs. The restore\_best\_weights option is also set to True, ensuring that the model reverts to the weights that achieved the lowest validation loss.

**Training Procedure:**

The training process utilizes the 'adam' optimizer and 'sparse\_categorical\_crossentropy' loss function. The 'accuracy' metric is used for performance evaluation. Input images are normalized by scaling pixel values to the range of [0, 1]. The training was halted early at the 6th epoch due to a lack of improvement in the validation loss, demonstrating the efficacy of the early stopping mechanism.

**Training Performance:**

The model demonstrated strong performance on the training data with an accuracy of approximately 97.69%, while the validation accuracy was approximately 96.01%. The early stopping reduced training time to approximately 155 seconds without compromising the model's ability to learn effectively from the training data.

**Model Evaluation:**

The "Base Model Greyscale with Early Stopping" underwent a thorough evaluation on the test set, which consisted of greyscale images. The performance metrics obtained are as follows:

* **Loss**: The model achieved a test loss of 0.1149, which indicates the model's prediction error on the test set.
* **Accuracy**: With an accuracy of 97.17%, the model demonstrated high precision in identifying the correct class labels for the test images. This suggests that the model can effectively distinguish between different distracted driving behaviors.

Figure 3 shows how loss and accuracy metrics for training and validation datasets vary throughout the training. It can be noticed that the model is trained only for 6 Epochs and still there is slight divergence in the validation loss from the training loss.

A screenshot of a graph

Description automatically generated

Figure 3: Training Metrics plot for Base Model using greyscale images and early stopping

## Base Model RGB:

The "Base Model RGB" is a convolutional neural network (CNN) optimized for processing RGB images. This model is designed to harness the full spectrum of color information available in the State Farm dataset's images to identify and classify distracted driver behaviors.

**Input:**

The model's input layer is designed to accept RGB images of size (120,160,3), indicating the images' height, width, and depth, respectively, with the depth of 3 representing the three-color channels: Red, Green, and Blue.

**Architecture:**

* **Convolutional Layers:**
  + The first layer is a 2D convolutional layer with 128 filters of size (3,3) and uses the 'relu' activation function to extract primary features from the input image.
  + Following this, a max pooling layer with a size of (2,2) reduces the spatial dimensions to 59×79 while retaining the essential features.
  + The second convolutional layer includes 64 filters with a kernel size (3,3), continuing the feature extraction process with 'relu' activation.
  + Another max pooling layer succeeds it, reducing the feature map size to 28×38.
  + The third convolutional layer has 32 filters with a kernel size (3,3). Again, it uses 'relu' activation to detect finer and more complex features in the input data.
  + A final max pooling layer downsizes the feature maps to 13×18, concluding the convolutional base of the model.
* **Flattening Layer:**
  + A flattening layer converts the 2D feature maps into a 1D feature vector with a size of 7488 elements, preparing the data for the dense layers.
* **Dense Layers:**
  + The network includes a dense layer with 512 neurons and 'relu' activation, serving as a fully connected layer to interpret the features extracted by the convolutional layers.
  + The output layer comprises 10 neurons corresponding to the 10 classes of driving behavior and uses a 'softmax' activation function to generate a probability distribution over the classes.

**Output:**

The model outputs a vector of size 10, with each element representing the probability that the input image belongs to one of the ten classified driver behaviors.

**Trainable Parameters:**

The model contains 3,935,338 trainable parameters, reflecting its capacity to learn detailed features from the complex, high-dimensional input data.

**Training Procedure:**

The model is compiled with the 'adam' optimizer and uses 'sparse\_categorical\_crossentropy' as the loss function. The accuracy metric is tracked during training. Image data is normalized by scaling pixel values between 0 and 1 to aid in efficient model convergence. The training procedure was completed over 30 epochs with a batch size 100, taking approximately 2301 seconds**.**

**Model Evaluation:**

Upon evaluation, the "Base Model RGB" achieved an accuracy of approximately 98.31% on the test dataset with a loss of 127.76. The model's performance underscores the utility of incorporating full-color depth in detecting and classifying distracted driving behaviors.

Figure 4 shows how loss and accuracy metrics for training and validation datasets vary throughout the training. Even in this case, the validation loss diverged from the training loss, indicating little overfitting.

A graph of a model

Description automatically generated with medium confidence

Figure 4: Training Metrics plot for Base Model using color (RGB) images

## RGB Model with Batch Normalization:

The "RGB Batch Normalization Model" is a refined convolutional neural network (CNN) that employs batch normalization to facilitate more efficient training and improve model generalization. This architecture is tailored to process RGB images, particularly enhancing learning stability and reducing internal covariate shifts.

**Input:**

The model processes RGB images with an input shape of (120,160,3), accounting for the images' height, width, and depth (color channels).

**Architecture:**

* **Convolutional Layers:**
  + The first layer is a 2D convolutional layer with 64 filters of size (3,3) and 'relu' activation, designed to capture the primary features from the input images.
  + Following the first convolutional layer, batch normalization is applied to standardize the activations, promoting faster and more stable training.
  + A max pooling layer with a size of (2,2) then reduces the spatial dimensions to 59×79.
  + The second convolutional layer contains 32 filters with a kernel size of (3,3), coupled with 'relu' activation, and is immediately followed by batch normalization.
  + This layer is also followed by a max pooling operation, reducing the feature map size to 28×38.
  + The third convolutional layer employs 24 size filters (3,3), 'relu' activation, and is again followed by batch normalization.
  + A final max pooling layer reduces the spatial dimensions to 13×18, concluding the convolutional portion of the network.
* **Flatten Layer:**
  + The output from the last pooling layer is transformed into a single 1D vector of size 5616 through a flattened layer, preparing the data for the fully connected layers.
* **Dense Layers:**
  + The flattened data is then fed into a dense layer with 128 neurons and 'relu' activation, which interprets the features and patterns learned by the convolutional base.
  + The final output layer consists of 10 neurons with 'softmax' activation, delivering a probability distribution over the 10 classes representing different driving behaviors.

**Output:**

The model generates a 10-element vector where each element signifies the probability that the input image corresponds to one of the ten classes.

**Trainable Parameters:**

The model has 747,698 trainable parameters, notably fewer than the previous models due to a more compact architecture that retains the essence of feature learning and classification.

**Batch Normalization:**

Batch normalization layers are interspersed after each convolutional layer and before the max polling layer. These layers normalize the input layer by adjusting and scaling the activations, thereby improving the training process and convergence rate.

**Training Procedure:**

The model utilizes the 'adam' optimizer and 'sparse\_categorical\_crossentropy' for the loss function. The model's accuracy is the primary metric during the training, and images are normalized by scaling pixel values to a [0, 1] range. The training is executed over 30 epochs with a batch size of 100 and completed in approximately 6089 seconds.

**Model Evaluation:**

The "RGB Batch Normalization Model" demonstrated excellent performance, with an accuracy of approximately 99.40% on the test set and a notably low loss of 0.0313. This level of accuracy suggests that batch normalization significantly contributes to the model's ability to generalize from the training data to unseen images.

Figure 5 shows how loss and accuracy metrics for training and validation datasets vary throughout the training. With the introduction of batch normalization, the RGB Base model improved very well. Loss and Accuracy curves show smooth training without divergence between training and validation data metrics.

A screenshot of a graph

Description automatically generated

Figure 5: Training Metrics plot for Base Model using color (RGB) images and batch normalization

## ResNet152:

The " ResNet152 (RGB)" model utilizes the pre-trained ResNet152 architecture as a base, leveraging transfer learning to capitalize on the features learned by this deep neural network on the ImageNet dataset. The ResNet152 model is well-known for its depth and performance in image classification tasks.

**Pre-Trained Base:**

* **ResNet152 Base:**
  + The ResNet152 base is loaded with weights pre-trained on the ImageNet dataset.
  + The base model is configured to exclude the top classification layers (include\_top = False) to allow for customization of the final layers for the specific task of distracted driver detection.
  + The input shape is set to (120,160,3), appropriate for the processed RGB images.
  + All layers in the pre-trained base are frozen (**layer.trainable = False**), meaning their weights will not be updated during training. This decision is crucial to preserve the learned features and reduce training time.

**Custom Top Layers:**

* **Flatten Layer:**
  + The output of the ResNet152 base is flattened to convert the 2D feature maps into a single long vector.
* **Dense Layers:**
  + A dense layer with 1000 neurons and 'relu' activation is added to the flattened output, serving as an interpretation layer for the features extracted by the ResNet base.
  + The final output layer is a dense layer with 10 neurons (one for each class of driving behavior) using a 'softmax' activation function to provide a probability distribution over the classes.

**Output:**

The model outputs a 10-dimensional vector with each element indicating the probability that the input image belongs to one of the ten classes representing specific distracted driving behaviors.

**Trainable Parameters:**

The custom top layers added to the ResNet152 base consist of 40,971,010 trainable parameters, while the non-trainable parameters from the base amount to 58,370,944. The considerable number of trainable parameters in the top layers gives the model a high capacity for learning from the specific driving behavior dataset.

**Training Procedure:**

The model is compiled with the 'adam' optimizer and 'sparse\_categorical\_crossentropy' as the loss function. It uses accuracy as the performance metric. During training, the model is fitted on the normalized RGB training images with corresponding labels for 30 epochs using a batch size of 100. The total training time taken is noted to be approximately 2021.56 seconds.

**Model Evaluation:**

Post-training, the ResNet model is evaluated on the test set, resulting in an accuracy of approximately 92.00% and a loss of 0.2916. This performance indicates the effectiveness of employing a pre-trained network for feature extraction and trainable layers tailored to the specific classification task.

Figure 6 shows how loss and accuracy metrics for training and validation datasets vary throughout the training. Even though there is no major divergence between training and validation dataset metrics, some irregularities can be noticed in the Loss and Accuracy curves indicating that it can be improved further.

A screenshot of a graph

Description automatically generated

Figure 6: Training Metrics plot for ResNet152 model

## VGG16:

The "VGG16 (RGB)" model adapts the VGG16 architecture, which is renowned for its simplicity and depth and has proven effective in various image classification tasks. The model incorporates the VGG16 network pre-trained on the ImageNet dataset, utilizing its rich feature extraction capabilities while tailoring the top layers to distracted driver behavior classification.

**Pre-Trained Base:**

* **VGG16 Base:**
  + The VGG16 base is initialized with weights from the ImageNet dataset.
  + Similar to the ResNet adaptation, the top classification layers of the VGG16 are not included (include\_top = False), allowing for adding custom layers specific to the classification of driving behaviors.
  + The input shape for the VGG16 base is set to (120,160,3), conforming to the resized RGB images in the dataset.
  + The weights of the VGG16 base layers are frozen to prevent changes during training, ensuring that the pre-trained feature representations are maintained.

**Custom Top Layers:**

* **Flatten Layer**
  + A flatten layer converts the 2D output tensor from the VGG16 base into a 1D tensor, allowing the subsequent dense layers to process the extracted features.
* **Dense Layers**
  + A dense layer with 1000 neurons and 'relu' activation is added to learn from the flattened features.
  + The final layer is dense, with 10 units corresponding to the different categories of distracted driving behaviors. It uses 'softmax' activation to produce the probability distribution over the classes.

**Output:**

The model outputs a 10-element vector, each corresponding to the likelihood of the input image belonging to a particular class of driver behavior.

**Trainable Parameters:**

The added custom dense layers bring the total number of trainable parameters to 7,691,010, providing a significant capacity for learning on top of the established VGG16 base and contributing an additional 14,714,688 non-trainable parameters.

**Training Procedure:**

The VGG16 adaptation is compiled with 'adam' as the optimizer and 'sparse\_categorical\_crossentropy' as the loss function, monitoring 'accuracy' as the performance metric. The training data, normalized to the range [0, 1], is passed through the model for 30 epochs with a batch size 100. The training time is recorded to be approximately 3177.95 seconds.

**Model Evaluation:**

Upon completion of training, the VGG16 model is evaluated. It achieves an impressive test accuracy of approximately 99.53% with a loss of 0.0205. This indicates that the model effectively leverages the pre-trained VGG16 architecture to classify driver behaviors accurately.

Figure 7 shows how loss and accuracy metrics for training and validation datasets vary throughout the training. Loss and Accuracy curves show smooth training without divergence between training and validation data metrics.

A screenshot of a graph

Description automatically generated

Figure 7: Training Metrics plot for VGG16 model

# Results:

The "State Farm Distracted Driver Detection" project was comprehensive in its approach to evaluating the performance of various deep learning models. The metrics that assessed each model's efficacy on the test dataset were accuracy, precision, recall, and F1-score. These metrics provide insight into not only the percentage of correct predictions (accuracy) but also the models' ability to minimize false positives (precision) and false negatives (recall), which are crucial in the context of distracted driver detection.

Model Performance Metrics:

* **Accuracy**
  + The RGB Batch Normalization and VGG16 models achieved perfect training accuracies (100%), indicating an excellent fit on the training data.
  + Regarding test accuracy, VGG16 led with an impressive 99.53%, followed closely by the RGB Batch Normalization model at 99.39%. The Base Model RGB also performed strongly, with a test accuracy of 98.31%.
  + Out of all models, the ResNet model has more difference between Training accuracy and validation accuracy.

A graph of a graph

Description automatically generated with medium confidence

Figure 8: Model wise Accuracies plot for Train, Validate and Test datasets

A graph with blue and white stripes

Description automatically generated

Figure 9 : Model wise Test Accuracies plot

|  |  |  |  |
| --- | --- | --- | --- |
| **model\_name** | **training\_accuracy** | **val\_accuracy** | **test\_accuracy** |
| base\_model\_greyscale | 0.99554038 | 0.981884062 | 0.98550725 |
| base\_model\_greyscale\_with\_es | 0.976935387 | 0.960144937 | 0.971683383 |
| base\_model\_rgb | 0.98919934 | 0.983277619 | 0.983054638 |
| rgb\_batch\_normalization | 1 | 0.98968786 | 0.993979931 |
| ResNet152 | 0.953452706 | 0.911649942 | 0.919955432 |
| VGG16 | 1 | 0.99386847 | 0.995317698 |

Table 1: Model wise Accuracies

* **Loss**
  + The VGG16 model exhibited the lowest test loss (0.0205), indicating its efficiency in classifying the test images with minimal error. The RGB Batch Normalization model also showed a low test loss (0.0313).

A graph with different colored bars

Description automatically generated

Figure 10 : Model-wise Loss plot for Train, Validate and Test datasets

|  |  |  |  |
| --- | --- | --- | --- |
| **model\_name** | **training\_loss** | **val\_loss** | **test\_loss** |
| base\_model\_greyscale\_with\_es | 0.088826716 | 0.309089392 | 0.114905752 |
| rgb\_batch\_normalization | 3.13E-05 | 0.048389602 | 0.031316418 |
| ResNet152 | 0.160658658 | 0.314087719 | 0.291597843 |
| VGG16 | 0.000283826 | 0.034424592 | 0.020529673 |

Table 2 : Model wise Losses

* **Class-Specific Metrics**
  + **Class 0: Safe Driving**

VGG16 demonstrated exceptional performance with a 99.89% accuracy and a 99.95% specificity, making it the premier choice for identifying safe driving behaviors. Its F1 score of 99.50% further confirms its reliability in this class.

* + **Class 1: Texting – Right**

VGG16 again stood out, achieving a perfect accuracy, specificity, and F1 score of 100%. This indicates its unparalleled capability to detect this form of distraction, which is crucial for interventions aimed at preventing texting while driving.

* + **Class 2: Talking on the Phone – Right**

The VGG16 model maintained its superior performance, with a 99.96% accuracy and a perfect specificity and F1 score. It underscores the model's effectiveness in distinguishing this subtle form of distraction.

* + **Class 3: Texting – Left**

RGB with Batch Normalization showed robust performance, closely followed by VGG16, suggesting that both models are highly capable of recognizing distractions associated with texting on the left hand.

* + **Class 4: Talking on the phone – Left**

Similar to Class 3, RGB with Batch Normalization and VGG16 provided excellent metrics, demonstrating their efficacy in detecting drivers talking on the phone with their left hand.

* + **Class 5: Operating the Radio**

VGG16's accuracy and specificity were again notably high, indicating its strong ability to identify drivers distracted by operating the radio, a common and risky behavior.

* + **Class 6: Drinking**

In this category, RGB with Batch Normalization showed a slight edge over other models, suggesting its potential usefulness in detecting drinking behaviors that divert attention from driving.

* + **Class 7: Reaching Behind**

VGG16 and RGB with Batch Normalization performed well, highlighting the challenge of detecting less frequent but dangerous behaviors such as reaching behind.

* + **Class 8: Hair and Makeup**

VGG16 exhibited strong performance, underscoring the importance of accurately identifying self-grooming behaviors that significantly distract drivers.

* + **Class 9: Talking to Passenger**

For detecting drivers engaged in conversation with passengers, VGG16 proved to be highly effective, as evidenced by its accuracy and F1 score.

The average of all F1 scores of all classes is calculated to identify the best model that predicts well in all cases. Figure 11 and Table 3 shows the Average F1 score of all classes for each model. Even this shows that VGG16 model surpassed all other models with an average of 0.9951. RGB Model with Batch Normalization stood next to it with an average of 0.9938

A graph with blue and white stripes

Description automatically generated

Figure 11: Model-wise Average F1 Score

|  |  |
| --- | --- |
| **model\_name** | **Avg\_f1** |
| base\_model\_greyscale | 0.985083 |
| base\_model\_greyscale\_with\_es | 0.971811 |
| base\_model\_rgb | 0.982819 |
| rgb\_batch\_normalization | 0.993858 |
| ResNet152 | 0.919202 |
| VGG16 | 0.995102 |

Table 3 : Model wise Average F1 Values

The main important thing of this use case is to ensure that the model detects distracted driving and alerts drivers if they are distracted. Hence the important aspect of the model should be able to have less number of false safe driving predictions (i.e. Less number of predictions saying the driver is driving safely even though the driver is distracted). Figure 12 and Table 4 show each model's false safe drive prediction rate using the test data. Even in this case VGG16 model excels very well with just 0.000446 rate.

A graph with blue bars and white text

Description automatically generated

Figure 12 : Model-wise false Safe driving prediction rate

|  |  |
| --- | --- |
| **model\_name** | **false\_safe\_drive\_prediction\_rate** |
| base\_model\_greyscale | 0.000892 |
| base\_model\_greyscale\_with\_es | 0.013601 |
| base\_model\_rgb | 0.003567 |
| rgb\_batch\_normalization | 0.001338 |
| ResNet152 | 0.027648 |
| VGG16 | 0.000446 |

Table 4 : Model-wise false Safe driving prediction rate

**Challenges Encountered:**

* **Overfitting and Gradient Vanishing**
  + Several models tended to either overfit the training data or diverge the loss values greatly due to gradient vanishing. This was mitigated by changing the number of layers, and nodes per layer, using early stopping, batch normalization, and using pre-trained models to ensure the models generalized well to unseen data.

**Implications of Findings:**

The findings of this project underscore the potential of using deep learning to detect distracted driving behaviors in real time. The high accuracy and F1 scores indicate that such models can be reliable when deployed in in-vehicle systems to alert drivers or monitor fleet management systems.

# Future Scope:

The results of this project suggest that further explorations into transfer learning models could yield even more robust systems for distracted driver detection. Fine-tuning the top layers of these models or exploring different architectures could further improve accuracy. Additionally, implementing real-time detection systems in vehicles could be a practical application of this research.

# Final Remarks:

The "State Farm Distracted Driver Detection" project demonstrates the potential of deep learning models in improving road safety. By accurately classifying various distracted driver behaviors. These models can form the basis of advanced driver assistance systems that warn drivers and mitigate the risks associated with distracted driving.

# Conclusions:

The application of transfer learning, mainly using the VGG16 model, resulted in the highest overall test accuracy and the lowest test loss, demonstrating the effectiveness of leveraging pre-trained networks for complex classification tasks. The RGB Batch Normalization model also showed superior performance, benefitting from enhanced training dynamics due to normalization.

Conversely, while the Base Model Greyscale with Early Stopping had the advantage of faster training, it did not match the high accuracy levels of the more sophisticated models, suggesting a trade-off between training efficiency and final performance.

# References:

1. [Kaggle State Farm distracted driver detection.](file:///Users/deepikadasaroju/Downloads/1.%09https:/www.kaggle.com/c/state-farm-distracted-driver-detection)
2. [World Health Organization (2020). Road traffic injuries.](file:///Users/deepikadasaroju/Downloads/1.%09https:/www.who.int/news-room/fact-sheets/detail/road-traffic-injuries)
3. [The Daily Star. (2020). 21 died on roads every day.](https://www.thedailystar.net/backpage/road-accident-in-bangladeh-21-died-every-day-1852867)
4. [Hesham M. Eraqi, Yehya Abouelnaga, Mohamed H. Saad, Mohamed N. Moustafa, "Driver Distraction Identification with an Ensemble of Convolutional Neural Networks"](https://www.hindawi.com/journals/jat/2019/4125865/)
5. [Dash, A.K. (.2019). Vgg16 architecture.](https://iq.opengenus.org/vgg16/)
6. [Machine Learning Mastery - CNN](https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/)
7. [Transfer Learning Art of fine tuning a pretrained model](https://www.analyticsvidhya.com/blog/2017/06/transfer-learning-the-art-of-fine-tuning-a-pre-trained-model/)
8. [RestNet Architecture](https://towardsdatascience.com/review-resnet-winner-of-ilsvrc-2015-image-classification-localization-detection-e39402bfa5d8)