Tight Risk Bounds for Distracted Driver Detection

Andres Gomez
University of Florida, College of
Electrical and Computer Engineering
Gainesville, FL
Andresggomez@ufl.edu

I. Introduction

The leading cause of fatal accidents in the U.S. is due to distracted driving. Although self-driving technologies are becoming commercially prevalent in the U.S., drivers are expected to be fully alert, prepared to take over at any moment. Many newer vehicles are equipped with driver alert systems, which monitor a driver's behavior either through data collected from external sensors or through a camera system and warn or alert the driver when unsafe behavior is detected. This work will apply and compare various deep learning models to the publicly available State Farm Distracted Driver Detection dataset as well as estimate tighter risk bounds of false positives for distracted drivers than would generally be obtained through confidence intervals. Risk bound characterizations, like binomial proportion confidence intervals, will be determined.

II. Background

In 2020, the National Highway Traffic Safety Administration reported 3,142 fatalities and an estimated 324,652 injuries from motor vehicle crashes involving distracted drivers. These fatalities not only include those within vehicles, but also nearby cyclists and pedestrians. That year, about 1 in 8 of all police-reported motor vehicle crashes were reported as distraction-affected crashes. The same report outlines that drivers between the ages of 15 and 20 are most at risk of being distracted at the time of a fatal crash [2].

With the emergence of higher processing power and affordable dashboard cameras, deep learning-based approaches for identifying distracted drivers have shown promise. Provided with 2D images or video stream, models can be trained on labeled data to classify distracted drivers from alert drivers. These models can be integrated into larger driver alert systems to warn drivers when unsafe behavior is being detected. Distractions while driving include cell phone usage, eating/drinking, communicating with passengers, adjusting hair/make-up, reaching behind, and interacting with radio or climate controls. Previous work has leveraged publicly available datasets to design and train a driver alert system. One such paper proposes a deep learning model-based system to classify drivers and alert distracted ones [3].

Previous work has yielded promising results applying deep learning techniques to the problem of identifying distracted drivers from 2D images. However, some works do not dive deeply into providing test set risk bounds of the classifier's performance. Moreover, these risk bounds are often obtained via Gaussian assumptions, which don't always hold up in practice. Provided the 2D images, this work will aim to create a high performing (>0.95 F1-score) binary classifier to identify distracted drivers and determine risk bounds free of Gaussian assumptions.

A. State Farm Distracted Driver Detection Dataset

Various publicly available datasets exist that can be used to train a proposed classifier. Abbas et al., introduces various datasets used to train various algorithms to classify distracted drivers [4]. Of those datasets, we have selected to use the State Farm Distracted Driver Detection dataset (SFDDD).

The SFDDD (https://www.kaggle.com/c/state-farm-distracted-driver-detection) consists of 2D images obtained by dashboard cameras. The dataset contains about 100,000 images, each a resolution of 640 by 840 pixels, spanning across 10 classes including normal driving, texting with the right or left hand, talking on the phone with the right or left hand, operating the radio, drinking, reaching behind, adjusting hair/makeup, and talking to the passengers. The training set consists of about 22,400 images, with distracted drivers making up just under 20,000 of the images.

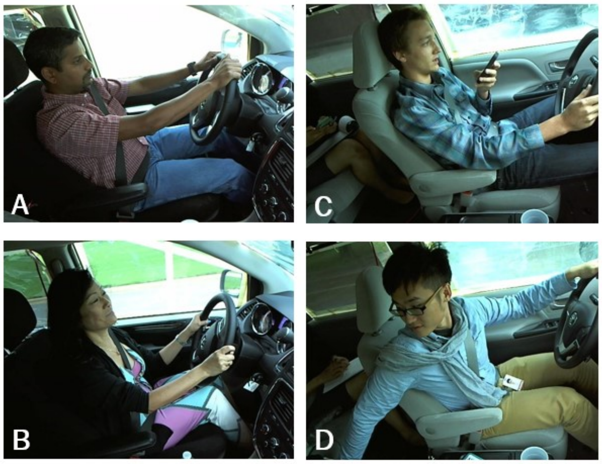


Figure 1: Sample images from the State Farm Distracted Driver Dataset containing alert drivers (A, B) and distracted drivers (C, D).

In *Figure 1*, we see sample images of alert and distracted drivers. Images A and B show drivers looking forward with both hands on the stearing wheel. Images C and D show drivers with a single hand on the stearing wheel. Image C shows the driver texting with their left hand while image D shows the driver reaching their right hand to the back seat while also looking backwards.

B. Deep Learning Approaches

Real-time detection of driver distraction has been achieved before using LSTM recurrent neural networks with accuracies of up to 96.6% [5]. Additionally, accuracies of

99% for the cause of distraction on the SFDDD dataset were obtained by Masood et al. through the implementation of VGG-16 [6]. Using the SFDDD dataset, Pal et al. perform multi-class classification with pretrained models like InceptionV3, ResNet, DenseNet, and MobileNet and obtain accuracies above 90%. Abbas et al. apply an ReSVM on the SFDDD and DrivFace datasets, outperforming state-of-the-art techniques, obtaining accuracies of 95.5% and 93.44% [4].

C. *Performance Evaluation and Error Bounds*

The main objective in this work is to identify distracted drivers. It is unsafe for a distracted driver to be falsely labeled as not distracted. Therefore, in an attempt to minimize the false positives (positive belonging to class not distracted), precision was selected to be the performance metric. Additionally, since the false negative rate is also important to consider, we will examine recall and the F1-score of the experiments.

To quantify the uncertainty estimate in precision, binomial proportion confidence intervals (BPCI) with 95% confidence will be determined rather than standard Gaussian-derived confidence intervals. The specific method used is the Clopper-Pearson interval, also called the exact method because its interval is obtained from the cumulative probabilities of the binomial distribution. This method is commonly used and is an alternative to using normal approximation. The bounds obtained by this method are depicted below.

$$\sum_{k=X}^{n} \binom{n}{k} p_L^k (1-p_L)^{n-k} = \alpha/2 \qquad \sum_{k=0}^{X} \binom{n}{k} p_U^k (1-p_U)^{n-k} = \alpha/2.$$

Figure 2: The equations for the lower and upper bounds obtained using the Clopper-Pearson interval.

This approach is taken because Gaussian assumptions do not always hold. Shah et al. propose modelling the risk instead as a binomial distribution. This proposed approach is analogous to providing a confidence interval around a binomial distribution [1]. In their empirical study, they obtained risk bounds through their binomial approach that were more realistic estimates on the limits compared to confidence intervals. Maximov et al. mention distribution-free characterization of risk bounds via VC dimension and data-dependent bounds known in terms of Rademacher complexities [7].

III. Methods

A. *Data Sampling and Augmentation*

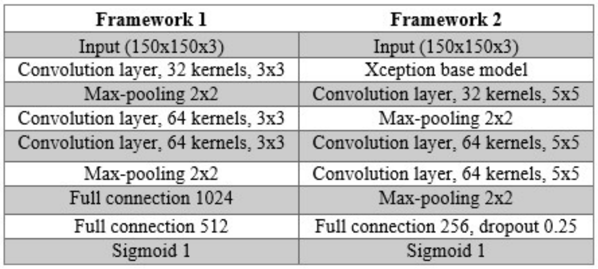
Due to RAM limitations and computational time, a sample of the original dataset set was selected. A sample of about 1130 images was selected for training/validation. This dataset was then split into a training and validation set by partitioning the data with an 80/20 split. To evaluate the accuracy of the estimated BPCI for the precision, 5 different tests sets, each containing approximately 225 images, were sampled from the original dataset. These test sets will be labeled A-E in later sections. All images were sampled from the original dataset without replacement.

To save additional memory and computational time, each RGB image was down sampled to a 150x150 image size. To account for noise in the data and enhance in generalizability, data augmentation was applied. The main transformation that was added were random rotations within a 0-to-45-degree range. This angle was decided upon after looking through the dataset and seeing that images were captured at varying angles.

B. *Convolution Neural Network Frameworks*

Two architectures, shown layer-by-layer in *Table 1*, were trained, and tested. The first network that was evaluated was one similar to the SCNNB, developed by Lei, Dai, and Ling (2020) [8]. The simplicity of the frameworks allows for faster training. The size of the input data is 150x150x3. As described in [8], SCNNB first extracts data features by a 3x3 convolution with 32 filters. Next, a 2x2 max-pooling layer is applied to reduce the dimensionality as well as the training time.

Afterwards, two sequential 3x3 convolution layers with 64 filters are applied to extract additional features. The output is fed to a 2x2 max-pooling layer. Then, a fully connected layer of 1280 neurons is applied, followed by another fully connected layer with 512 neurons. In the original SCNNB, the last layer is a softmax output layer and is used to achieve multi-classification. Since this problem is a binary classification problem, the last layer was changed to a single neuron with a sigmoid activation function.



Framework 1	Framework 2
Input (150x150x3)	Input (150x150x3)
Convolution layer, 32 kernels, 3x3	Xception base model
Max-pooling 2x2	Convolution layer, 32 kernels, 5x5
Convolution layer, 64 kernels, 3x3	Max-pooling 2x2
Convolution layer, 64 kernels, 3x3	Convolution layer, 64 kernels, 5x5
Max-pooling 2x2	Convolution layer, 64 kernels, 5x5
Full connection 1024	Max-pooling 2x2
Full connection 512	Full connection 256, dropout 0.25
Sigmoid 1	Sigmoid 1

Table 1: Architectures of the two CNNs that were trained and evaluated. Framework 1 corresponds to the CNN trained from scratch. Framework 2 utilizes transfer learning techniques.

C. *Transfer Learning with Xception*

Framework 2 had the Xception model as the base model. The Xception model, developed by Francois Chollet in 2016, is a leading image classification CNN. In a competition involving over 350 million photos with 17,000 classes, it dramatically outperformed the previous competition leader. The main difference between the Xception model and other leading CNN architectures is it assumes cross-channel and spatial patterns can be analyzed separately. So, the architecture first separates the channels, identifies spatial features from each channel, then applies depth-wise 1x1 convolution to look for cross-channel patterns amongst the extracted features [9]. These depth-wise convolutions cannot caprute spatial patterns, since they are of size 1x1. Hence, their role is to primarily extract deppthwise information from the spatial information extracted across each channel.

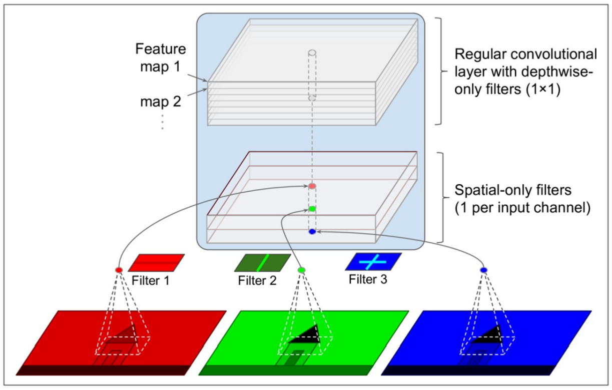

Feature map 1
map 2
Regular convolutional layer with depthwise-only filters (1×1)
Spatial-only filters (1 per input channel)
Filter 1
Filter 2
Filter 3

Figure 3: Diagrammatic representation of the Xception architecture.

After including the imagenet weights and discarding the top layer, an architecture very similar to framework 1 was constructed on top. The layer that follows the Xception base model is a 32 filter, 5x5 convolution layer, followed by a 2x2 max-pooling layer. Two sequential 5x5 convolution layers with 64 filters are applied, followed by a 2x2 max-pooling layer. The last layer is a fully connected layer of 256 neurons followed by a 0.25 dropout. The last layer is a single neuron with a sigmoid activation function.

The loss function for both frameworks was binary cross entropy, with an Adam optimizer with a learning rate of 0.0005. As previously mentioned, the performance metric was precision. Both frameworks were tested with 100 epochs, a patience of 20, and a batch size of 64.

IV. Training Evaluation

A. Framework Comparison

Initially, the two frameworks presented in *Table 1*, were evaluated on the validation set. *Table 2* shows the F1-score and BPCI at 95% confidence for the precision obtained by each framework on the validation set. We see that Framework 2 obtained a higher F1-score as well as tighter risk bounds. Hence, the remaining results will be carried out for Framework 2 only.

Metric	**Model**	
	Framework 1	Framework 2
F1-Score	0.95	1.0
95% BPCI Precision	[6.54, 14.8]	[0.00, 1.62]

Table 2: F1-score and BPCI of the precision at 95% confidence obtained by each Framework on the validation set.

V. ResuLts

We evaluated the accuracy of the estimated 95% BPCI for the precision on each of the test sets, labeled A-E in later sections. Additionally, we tracked the precision, recall and F1-score for each of the test sets. We can see from Table 3, that the 95% BPCI was fairly consistent on all of the test sets. The lower bound of the 95% BPCI for precision ranged between 0.31 and 0.79, and the upper bound ranged between 4.26 and 5.57. The values for precision, recall, and the F1-score were all consistent as well.

Test Set	**95% BPCI Precision**	**Precision**	**Recall**	**F1-score**
A	[0.32, 4.41]	0.985	0.946	0.96
B	[0.79, 5.57]	0.976	0.980	0.98
C	[0.79, 5.55]	0.976	0.985	0.98
D	[0.31, 4.26]	0.985	0.980	0.98
E	[0.54, 5.02]	0.980	0.961	0.97

Table 3: The 95% BPCI for precision, the precision, recall and F1-score obtained by Framework 2 on each of the 5 tests sets.

VI. Conclusion

Two CNN architectures were trained on a sample of the State Farm Distracted Driver Detection dataset to identify distracted drivers. The first framework was inspired by the shallow convolutional neural network, presented by Lei et al. The second framework utilized Xception as the base model, followed by convolution and fully connected layers. Additionally, binomial proportion confidence intervals were obtained to determine the true proportion of false positives, defined as distracted drivers falsely labeled as not distracted.

As outlined in *Table 2*, framework 2 outperformed framework one in terms of F1-score and tighter binomial proportion confidence intervals were also obtained for framework 2. On 5 separate test sets, the F1-score ranged between 0.96 and 0.98, the recall ranged between 0.961 and 0.985, and the Precision ranged between 0.976 and 0.985. Additionally, the 95% BPCI for precision remained consistent amongst all of the individual test sets. The lower bound ranged between 0.31 and 0.79, and the upper bound ranged between 4.26 and 5.57.

VII. References

[1] M. Shah and S. Shanian, “Hold-out Risk Bounds for Classifier Performance Evaluation,” 2009.

[2] N. Highway Traffic Safety Administration and U. Department of Transportation, “Distracted Driving 2020,” 2020.

[3] A. Pal, S. Kar, and M. Bharti, “Algorithm for Distracted Driver Detection and Alert Using Deep Learning,” Optical Memory and Neural Networks (Information Optics), vol. 30, no. 3, pp. 257–265, Jul. 2021, doi: 10.3103/S1060992X21030103.

[4] T. Abbas et al., “Deep Learning Approach Based on Residual Neural Network and SVM Classifier for Driver’s Distraction Detection,” Applied Sciences (Switzerland), vol. 12, no. 13, Jul. 2022, doi: 10.3390/app12136626.

[5] M. Wöllmer et al., “On-line Driver Distraction Detection using Long Short-Term Memory.”

[6] S. Masood, A. Rai, A. Aggarwal, M. N. Doja, and M. Ahmad, “Detecting distraction of drivers using Convolutional Neural Network,” Pattern Recognit Lett, vol. 139, pp. 79–85, Nov. 2020, doi: 10.1016/J.PATREC.2017.12.023.

[7] Y. Maximov and D. Reshetova, “Tight risk bounds for multi-class margin classifiers,” Pattern Recognition and Image Analysis, vol. 26, no. 4, pp. 673–680, Oct. 2016, doi: 10.1134/S105466181604009X.

[8] Lei, F., Liu, X., Dai, Q. Shallow convolutional neural network for image classification. SN Appl. Sci. 2, 97 (2020).

[9] F Chollet. “Xception: Deep Learning with Depthwise Separable Convolutions” 2017.