Scientific Journal of Impact Factor (SJIF): 4.72

e-ISSN (O): 2348-4470
p-ISSN (P): 2348-6406

International Journal of Advance Engineering and Research Development

Volume 4, Issue 4, April -2017

# Driver Distraction Detection System

Prof.Nilima Walde[1], Kartik Singh Chauhan[2], Akanchha Singh Bhadauria[3], Deepak Kumar Singh[4], Abhishek Ray[5]

[1,2,3,4,5]*Department of Information Technology, Army Institute of Technology, Pune*

**Abstract** — *As we know one of the major causes of accident is distracted states of driver thus our paper focuses on solving this problem by applying machine learning technique. We aim at building a model using Convolutional Neutral Network which could classify the images based on different states of driver. This model could be used in real time to acquaint the driver about his wrong driving deeds and prevent accidents due to distracted states. The images that are used for training and building the model has been taken using a static driving simulator with real human subjects performing a specific secondary task. The distracted states have been categorized under 10 different categories which include texting, talking to passenger, talking on the phone, looking left or right, searching something at the back seat, hair and makeup, operating radio and drinking.*

***Keywords***- *Driver distraction, machine learning, convolutional neural network, accident prevention, maxpooling*.

## I. INTRODUCTION

DISTRACTED DRIVING is any activity that could divert a person's attention away from the primary task of driving. All distractions endanger driver, passenger, and bystander safety. These types of distractions include: Texting Using a cell phone or smartphone Eating and drinking Talking to passengers Grooming Reading, including maps Using a navigation system Watching a video Adjusting a radio, CD player, or MP3 player in 2014, 3,179 people were killed, and 431,000 were injured in motor vehicle crashes involving distracted drivers. Ten percent of all drivers 15 to 19 years old involved in fatal crashes were reported as distracted at the time of the crashes. This age group has the largest proportion of drivers who were distracted at the time of the crashes. (NHTSA) The percentage of driver's text-messaging or visibly manipulating handheld devices increased from 1.7 percent in 2013 to 2.2 percent in 2014. Since 2007, young drivers (age 16 to 24) have been observed manipulating electronic devices at higher rates than older drivers. As we know one of the major causes of accident is distracted states of driver thus our study focuses on solving this problem by applying machine learning technique.

We aim at building a model using Convolutional Neutral Network which could classify the images based on different states of driver. This model could be used in real time so as to acquaint the driver about his wrong driving deeds and prevent accidents due to distracted states. The images that are used for training and building the model has been taken using a static driving simulator with real human subjects performing a specific secondary task. The distracted states have been categorized under 10 different categories which include texting, talking to passenger, talking on the phone, looking left or right, searching something at the back seat, hair and makeup, operating radio and drinking. The increasing number of fatalities demonstrates that driving safety represents a persistent and important issue. Reducing crash involvement would benefit millions of people across the world. Although most motor-vehicle crashes are attributed to multiple causes, driver error represents a dominant one because drivers are responsible for operating vehicles and avoiding crashes (Lee, 2006). Compared to 34.9 vehicle factors, driver errors contribute to 92.9 example, rear-end collisions that comprise approximately 30 departure crashes, which cause the greatest number of fatalities have been largely attributed to the inability of drivers to detect hazards and control the vehicle properly (The National Safety Council, 1996). Most of these performance breakdowns result from the impairments of driver's attention. Any secondary task that is not primarily related to driving may cause distraction to drivers and deteriorate driver's ability to perform driving related tasks. Lee et al. (2008) followed the later 6 approach and define distracted driving as, diversion of attention away from activities critical for safe driving toward a competing activity. Based on these definitions, it can be concluded that any tasks irrelevant to driving that reduces the situational awareness of a driver is some kind of distraction. Machine learning (ML) technology may be able to provide the right algorithms for coping with such a challenge. ML is the technique of searching large volumes of data for unknown patterns. It has been successfully applied in business, health care, and other domains [2], [3].

In particular, this technology can be applied to build a discrimination model that captures the differences in behavior when people drive normally and when they are distracted. This paper is organized as follows. Section II provides the definition of driver distraction based on the current discussion in the literature. Then, Section III briefly describe the different driver state classes. The experimental setup will be shown in Section IV, whereas Section V deals with the procedure and implementation. Section VI and VII describes the results and future work respectively. Finally, Section VIII concludes this paper with a summary of the main points of interest in this research.

 

## II. DEFINITION OF DISTRACTED DRIVING

Several definitions very often overlapped and mixed with inattention or with other driver's states, such as drowsiness and workload. For what concerns the definition of distraction adopted in this paper, we have considered the taxonomy proposed by Regan et al. [1] and by Lee et al. [4]. In particular, we start from the following definition: Driver distraction is the diversion of attention away from activities critical for safe driving toward a competing activity. This has been extended by Regan et al., adding the concept of driver inattention, which means insufficient or no attention to critical activities for safe driving toward a competing activity. To sum up, distinct from other forms of driver inattention, distraction occurs when a driver's attention is diverted away from driving by a secondary task that requires focusing on an object, an event, or a person not related to the driving task. Although existing data are inadequate and not representative of the driving population, it is estimated that drivers engage in potentially distracting secondary tasks approximately 30time that their vehicles are in motion. (Having a conversation with passengers is the most frequent secondary task, followed by eating, smoking, manipulating controls, reaching inside the vehicle, and using cell phones). Thus, we have considered visual distraction as the diversion of visual attention away from the road. This category of driver distraction is also the one used by Lee et al. [4].

## III. DISTRACTED STATE CLASSES

The 10 classes to predict are:

- c0: safe driving
- c1: texting - right
- c2: talking on the phone - right
- c3: texting - left
- c4: talking on the phone - left
- c5: operating the radio
- c6: drinking
- c7: reaching behind
- c8: hair and makeup
- c9: talking to passenger

*Figure 1. Distracted states mentioned above*

## IV. EXPERIMENTAL SETUP

The installation required for this study involves:

- Anaconda is the leading open data science platform powered by Python. The open source version of Anaconda is a high-performance distribution of Python and R and includes over 100 of the most popular Python, R and Scala packages for data science
- Keras is a high-level neural networks API, written in Python and capable of running on top of either TensorFlow or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.
- OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

 

- Theano is a Python library that allows you to define, optimize, and evaluate mathematical expressions involving multi-dimensional arrays efficiently. Theano features: tight integration with NumPy Use numpy, ndarray in Theano-compiled functions. transparent use of a GPU Perform data-intensive computations much faster than on a CPU. efficient symbolic differentiation Theano does your derivatives for functions with one or many inputs. speed and stability optimizations Get the right answer for log(1+x) even when x is really tiny. dynamic C code generation Evaluate expressions faster. extensive unit-testing and self-verification Detect and diagnose many types of errors. Theano has been powering large-scale computationally intensive scientific investigations since 2007. But it is also approachable enough to be used in the classroom (University of Montreals deep learning/machine learning classes).
- The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and explanatory text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, machine learning and much more.

## V. PROCEDURE AND IMPLEMENTATION

Convolutional Neural Networks are very similar to ordinary Neural Networks as they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SoftMax) on the last (fully-connected) layer and all the tips/tricks we developed for learning regular Neural Networks still apply. In order to develop a convolutional model, we must know certain hyperparameters involved for building the model. The hyperparameters used by us involve:

- Number of features
- Size of features
- Window Size
- Window Stride
- Fully Connected
- Window Size

The keras library provides us the way to build the layers instead of working with the neurons. The epoch, batch size and number of filters must be decided by the programmer. Neurons in a fully connected layer have full connections to all activations in the previous layer, as seen in regular Neural Networks. Their activations can hence be computed with a matrix multiplication followed by a bias offset. The most common form of a ConvNet architecture stacks a few CONV-RELU layers, follows them with POOL layers, and repeats this pattern until the image has been merged spatially to a small size. At some point, it is common to transition to fully-connected layers. The last fully-connected layer holds the output, such as the class scores.

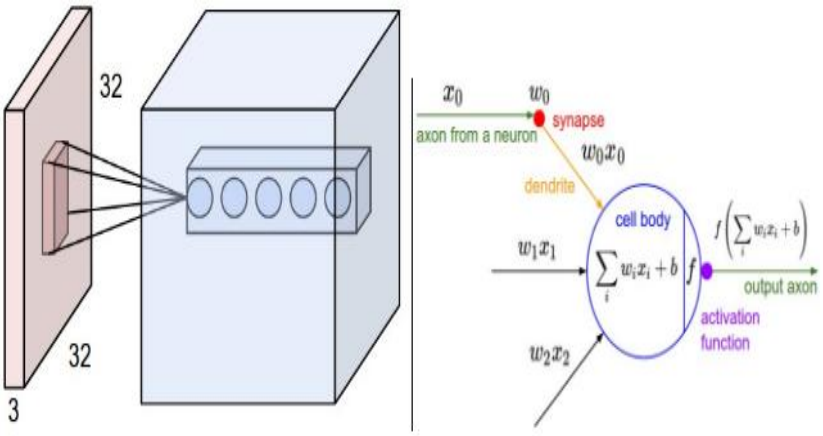


*Figure 2. Convolutional Neural Network*

The core data structure of Keras is a model, a way to organize layers. The simplest type of model is the Sequential model, a linear stack of layers. For more complex architectures, you should use the Keras functional API, which allows to build arbitrary graphs of layers.

 

```
model = Sequential()
```

Stacking layers is as easy as `.add()`:

```
from keras.layers import Dense, Activation

model.add(Dense(units=64, input_dim=100))
model.add(Activation('relu'))
model.add(Dense(units=10))
model.add(Activation('softmax'))
```

Once your model looks good, configure its learning process with `.compile()`:

```
model.compile(loss='categorical_crossentropy',
              optimizer='sgd',
              metrics=['accuracy'])
```

*Figure 3. Keras stacking of layers*

Another important concept of CNNs is pooling, which is a form of nonlinear down-sampling. There are several non-linear functions to implement pooling among which max pooling is the most common. It partitions the input image into a set of nonoverlapping rectangles and, for each such sub-region, outputs the maximum. The intuition is that the exact location of a feature is less important than its rough location relative to other features. The pooling layer serves to progressively reduce the spatial size of the representation, to reduce the number of parameters and amount of computation in the network, and hence to also control overfitting. It is common to periodically insert a pooling layer between successive convolutional layers in a CNN architecture. The pooling operation provides a form of translation invariance.

The parameters that are assigned while building this model includes these values in Figure 4. Convolutional neural networks are often used in image recognition systems. They have achieved an error rate of 0.23 percent on the MNIST database, which as of February 2012 is the lowest achieved on the database. Another paper on using CNN for image classification reported that the learning process was" surprisingly fast"; in the same paper, the best published results at the time were achieved in the MNIST database and the NORB database. Thus, we aimed at building the model using CNN. This model overall took around 6-8 hours to build using the training set but surely with good GPU processing capabilities the efficiency of the model can be increased.

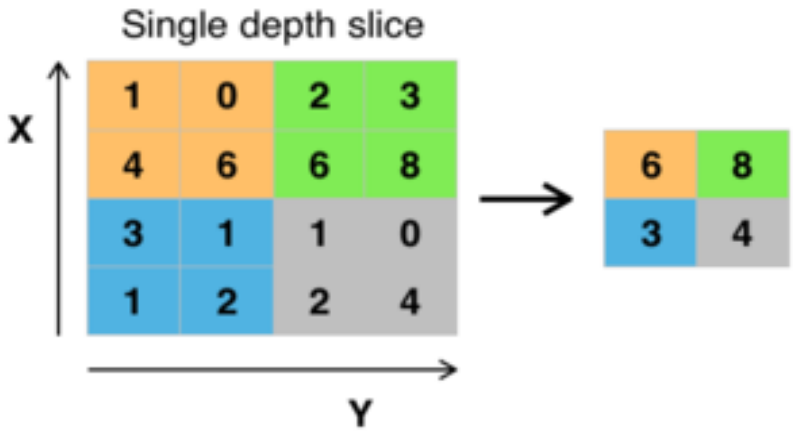


*Figure 4. Max-pooling in cnn*

 

```
batch_size = 64
nb_classes = 10
nb_epoch = 2
# input image dimensions
img_rows, img_cols = 96, 128
# number of convolutional filters to use
nb_filters = 32
# size of pooling area for max pooling
nb_pool = 2
# convolution kernel size
nb_conv = 3
```

*Figure 5. Hyperparameters for our experiment*

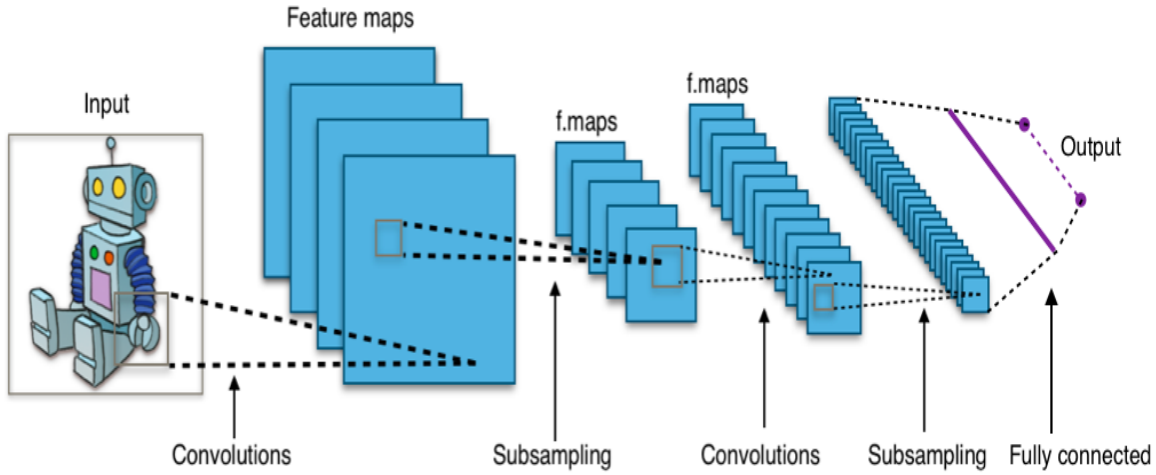


*Figure 6. Overall Architecture of CNN*

## VI. RESULTS

The model provides an accuracy score of .63 which is quite good when compared to building a model with SVM or other classifiers. That is out of 10 images our model could predict right answers for at least 6 images. The score for each of the classes for a particular image from the testing set has been stored in a csv file.

| | A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | c0 | c1 | c2 | c3 | c4 | c5 | c6 | c7 | c8 | c9 | c10 |
| 2 | 0.002900028 | 0.001493741 | 0.011507381 | 0.00757677 | 0.003434383 | 0.907077253 | 0.004256526 | 0.05000808 | 0.00240537 | 0.0093405 | img_1.jpg |
| 3 | 0.054595083 | 0.0236168 | 0.02242822 | 0.24033606 | 0.11552719 | 0.252046496 | 0.017762592 | 0.06742712 | 0.0325904 | 0.17367 | img_10.jpg |
| 4 | 0.427710682 | 0.258426875 | 0.033833295 | 0.01737818 | 0.072916135 | 0.001814132 | 0.022391567 | 0.0915503 | 0.05350773 | 0.0204711 | img_100.jpg |
| 5 | 0.20709163 | 0.199017137 | 0.21774447 | 0.00185473 | 0.008231587 | 0.000766106 | 0.163181499 | 0.01637148 | 0.16269439 | 0.023047 | img_1000.jpg |
| 6 | 0.175210372 | 0.012817558 | 0.01275373 | 0.31079972 | 0.193856522 | 0.039003082 | 0.014804712 | 0.02128631 | 0.03555393 | 0.1839141 | img_100000.jpg |
| 7 | 0.066754185 | 0.004148643 | 0.011661227 | 0.38791528 | 0.246790305 | 0.091440424 | 0.00732243 | 0.00449907 | 0.01291451 | 0.1665539 | img_100001.jpg |
| 8 | 0.558428407 | 0.200665295 | 0.044377107 | 0.00596246 | 0.105209179 | 0.00044312 | 0.008159002 | 0.0180533 | 0.04710896 | 0.0115931 | img_100002.jpg |
| 9 | 0.082979806 | 0.401007831 | 0.161996543 | 0.01900433 | 0.076937504 | 0.03402029 | 0.032946624 | 0.09416454 | 0.05413698 | 0.0428055 | img_100003.jpg |
| 10 | 0.097132497 | 0.162630737 | 0.177345708 | 0.03061308 | 0.209266275 | 0.023033135 | 0.056169502 | 0.04272766 | 0.08698128 | 0.1141001 | img_100004.jpg |
| 11 | 0.28164953 | 0.090358436 | 0.017778689 | 0.09481884 | 0.347082287 | 0.002985022 | 0.021458311 | 0.09659841 | 0.02591923 | 0.0213512 | img_100005.jpg |
| 12 | 0.100105152 | 0.063773118 | 0.240964964 | 0.0149455 | 0.127637669 | 0.008759942 | 0.030784363 | 0.06704995 | 0.2473812 | 0.0985981 | img_100007.jpg |
| 13 | 0.098371975 | 0.044518836 | 0.018855775 | 0.00045875 | 0.0069078 | 0.000130684 | 0.027039766 | 0.10871931 | 0.66447854 | 0.0305186 | img_100008.jpg |
| 14 | 0.076186828 | 0.120338187 | 0.454713255 | 0.00365128 | 0.028718118 | 0.004026515 | 0.059574679 | 0.0205585 | 0.1888504 | 0.0433822 | img_100009.jpg |
| 15 | 0.317504793 | 0.043602373 | 0.042683765 | 0.0529712 | 0.051424917 | 0.027594516 | 0.037436631 | 0.03776096 | 0.17071207 | 0.2183088 | img_10001.jpg |
| 16 | 0.284323633 | 0.064432383 | 0.229552388 | 0.00126 | 0.004125459 | 0.001377248 | 0.118908249 | 0.00523022 | 0.25484163 | 0.0359488 | img_100010.jpg |
| 17 | 0.225687832 | 0.078612857 | 0.057668008 | 0.15373568 | 0.270010233 | 0.010958361 | 0.060918421 | 0.06245244 | 0.02806059 | 0.0518956 | img_100011.jpg |
| 18 | 0.083719239 | 0.121534482 | 0.080285802 | 0.06169575 | 0.067154378 | 0.089306034 | 0.14590548 | 0.27195457 | 0.0334818 | 0.0449625 | img_100012.jpg |
| 19 | 0.177521318 | 0.02478037 | 0.011868466 | 0.26886106 | 0.259722948 | 0.04115985 | 0.017438287 | 0.03731146 | 0.04665904 | 0.1146772 | img_100013.jpg |
| 20 | 0.183497593 | 0.246879488 | 0.289851844 | 0.01052741 | 0.053372037 | 0.003469787 | 0.055687286 | 0.02082746 | 0.09986877 | 0.0360183 | img_100014.jpg |

submission_0.642711_2017-02-12-

*Figure 7. CSV File of scores for test set*

 

## VII. FUTURE WORK

Once the distracted state of the driver is detected through our model we could use reinforcement learning in the car which would take autonomous control of the car to prevent accidents.

## VIII. CONCLUSION

In this paper, we have tried to present an overview of different driver distraction classifiers based on ML techniques. Our study will predict all sorts of distracted state of the driver and thus this application could be used in order to acquaint driver for wrong driving deeds which could finally help in accident prevention. This paper can be implemented in developing systems that would be able to alert the driver if found distracted and help in preventing the accident. This paper works on classifying images but it could be extended for real time monitoring application that could detect a distracted state in a video feed and generate an alarm.

## REFERENCES

[1] M. A. Regan, C. Hallet, and C. P. Gordon, Driver distraction and driver inattention: Definition, relationship and taxonomy, Accid. Anal. Prev. J.,vol. 43, no. 5, pp. 17711781, Sep. 2011.
[2] P.-N. Tan, Introduction to Data Mining. Pearson. Boston, MA, USA: Addison-Wesley, 2005.
[3] P. BaldiandS.Brunak, Bioinformatics: TheMachineLearningApproach, 2nd ed. Cambridge, MA, USA: MIT Press, 2001.
[4] J. D. Lee, K. L. Young, and M. A. Regan, Dening driver distraction, in Driver Distraction: Theory, Effects, and Mitigation, M. A. Regan, J. D. Lee, and K. L. Young, Eds. Boca Raton, FL, USA: CRC, 2008, pp. 3140.

 