Drive-Net: Convolutional Network for Driver Distraction Detection

Mohammed S. Majdi†, Sundaresh Ram*, Jonathan T. Gill†, Jeffrey J. Rodríguez†
†Department of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, USA
*School of Electrical and Computer Engineering, Cornell University, Ithaca, NY, USA
Email: {mohammadsmajdi, jtgill, jjrodrig}@email.arizona.edu, sr2255@cornell.edu

***Abstract*—To help prevent motor vehicle accidents, there has been significant interest in finding an automated method to recognize signs of driver distraction, such as talking to passengers, fixing hair and makeup, eating and drinking, and using a mobile phone. In this paper, we present an automated supervised learning method called Drive-Net for driver distraction detection. Drive-Net uses a combination of a convolutional neural network (CNN) and a random decision forest for classifying images of a driver. We compare the performance of our proposed Drive-Net to two other popular machine-learning approaches: a recurrent neural network (RNN), and a multi-layer perceptron (MLP). We test the methods on a publicly available database of images acquired under a controlled environment containing about 22425 images manually annotated by an expert. Results show that Drive-Net achieves a detection accuracy of 95%, which is 2% more than the best results obtained on the same database using other methods.**

***Index Terms*—Image classification, convolutional neural networks, random forest, driver distraction.**

I. Introduction

Distracted driving is a major cause of motor vehicle accidents. Each day in the United States, approximately 9 people are killed and more than 1000 are injured in crashes that involve a distracted driver [1]. It is estimated that roughly 25% of motor vehicle accident fatalities are due to distracted driving [2]. A study of American motor vehicle fatalities [1] reveals the top 10 causes of distracted driving:

1) Generally distracted or "lost in thought - 62%
2) Mobile phone use - 12%
3) Outside person, object, or event – 7%.
4) Other occupants - 5%.
5) Using or reaching for a device brought into the car (e.g., phone) 2%.
6) Eating or drinking - 2%.
7) Adjusting audio or climate controls - 2%.
8) Using devices/controls to operate the vehicle (e.g., adjusting mirrors or seatbelts) - 1%.
9) Moving objects (e.g., insects or pets) - 1%
10) Smoking related - 1%.

Therefore, there is an interest in using analysis of dashboard camera images to automatically detect drivers engaged in distracting behavior. A dataset of such dashboard camera images, observing various activities of drivers, has been compiled and used for the Kaggle competition regarding automated detection of driver distraction [3]. Fig. 1 shows examples of a few dashboard camera images manually annotated as different activities when driving.

Fig. 1. Some representative dashboard camera images of drivers being distracted. *From left to right:* drinking while driving, safe driving, reaching behind when driving, talking on the phone (left hand) while driving [3].

Object detection and human behavior detection are well-researched topics in the computer vision literature [4]–[8]. Machine learning (esp. deep learning) techniques can often learn complex models and achieve high accuracy, so many researchers have started to apply such techniques to solve computer vision problems including object detection and human behavior detection. For example, the Inception-v4 model proposed by Szegedy *et al.* [9] is a supervised learning model made up of deep convolutional residual networks (ResNet) that has more than 75 trainable layers, and it has been shown to achieve 96.92% accuracy on the ImageNet dataset. Girshick *et al.* [10] introduced a very powerful method for object detection and segmentation using a region-based convolutional neural network (CNN). This method divides the human behavior detection problem into two problems. First, they apply an object detection algorithm to detect the regions of interest (ROI) where people are present within an image. Next, each ROI is fed to a CNN to identify the type of behavior exhibited in the given ROI. Adding other traditional machine learning methods such as ensemble learning (i.e., bagging), and K nearest neighbors (KNN) to the CNN model is a way of improving the accuracy of the already existing model [11].

One of the main drawbacks of CNNs is that training the network using a large dataset can lead to over-fitting the model. In order to avoid this, ensemble methods such as random decision forests can be effective. With this in mind, we propose a new supervised learning algorithm called Drive-Net that combines a CNN and a random forest in a cascaded fashion for application to the problem of driver distraction detection using dashboard camera images. We compare our proposed Drive-Net to two other neural network methods: a residual neural network (RNN) and a multi-layer perceptron (MLP). We show that Drive-Net achieves better classification accuracy than the

Authorized licensed use limited to: Sterne Kessler Goldstein Fox. Downloaded on November 05,2025 at 17:07:21 UTC from IEEE Xplore. Restrictions apply.

Samsara EX1013
Samsara v. Motive
IPR2026-00108

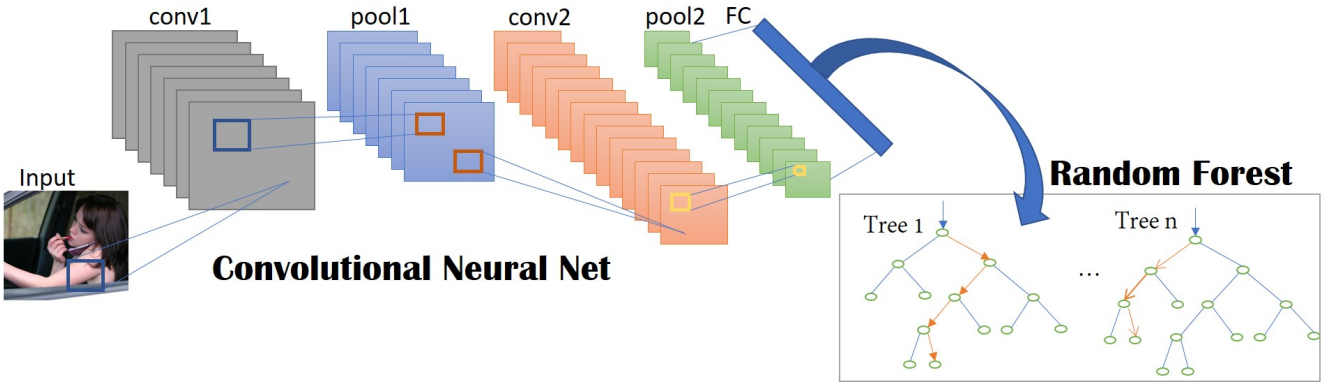

Fig. 2. An overview of the proposed Drive-Net. Our proposed CNN architecture (shown on the left side) consists of two convolution layers (conv) each followed by a maxpooling layer (pool) and a final ReLU layer, the output of which is regularized using dropouts to obtain a fully connected layer (FC). The FC layer is fed as input to the random forest classifier (on the right side), which predicts the final class label.

driver distraction detection algorithms that were proposed in the Kaggle competition [3].

II. Methods

Our proposed method, Drive-Net, is a cascaded classifier consisting of two stages: a CNN as the first stage, whose output layer is fed as the input to a random decision forest to predict the final class label. We define each stage in detail below.

A. Convolutional Neural Network Configuration

We adopt the U-Net architecture [12] as the basis of our CNN. The motivation behind this architecture is that the contracting path captures the context around the objects in order to provide a better representation of the object as compared to architectures such as Alexnet [15] and VGGnet [16]. Very large networks like Alexnet and VGGnet require learning a massive number of parameters and are very hard to train in general, needing significant computational time. Thus, we empirically modify the U-Net architecture in this work to suit our application.

To construct our CNN, we discard U-Net's layers of up-convolution and the last two layers of down-sampling and replace them with a 1×1 convolution instead to obtain a fully connected layer. We use the rectifier activation function [12] for our CNN as the constant gradient of rectified linear units (ReLUs) results in faster learning and reduces the problem of vanishing gradient compared to hyperbolic tangent (tanh). We implement a maxpooling layer instead of average pooling in the sub-sampling layer [15]. We observed that the performance is better when a ReLU layer was configured with maxpooling layer, resulting in higher classification accuracy after 50 epochs. We used the 1×1 convolutional filters for the Adam [13] optimizer. All the other parameters such as number of layers, convolutional kernel size, training algorithm, and the number of neurons in the final dense layer were all experimentally determined for our application.

In order to keep the training time small, we reduce the size of dashboard camera images by a factor of 10 making them 64×48 in size and feed these as the input to our CNN. We do not zero-pad the image patches, as the ROI with human activity is located towards the center of the image. Two consecutive convolutional layers are used in the network. The first convolutional layer consists of 32 kernels of size $5 \times 5 \times 1$. The second convolutional layer consists of 64 kernels of size $5 \times 5 \times 32$. The sub-sampling layer is set as the maximum values in non-overlapping windows of size 2×2 (stride of 2). This reduces the size of the output of each convolutional layer by half. After the two convolutional and sub-sampling layers, we use a ReLU layer, where the activation y for a given input x is obtained as

$$y = f(x) = \max(0, x) \tag{1}$$

A graphical representation of the architecture of the proposed CNN model is shown in Fig. 2 (see the left side).

B. Random Decision Forest

A random forest classifier consists of a collection of decision tree classifiers combined together to predict the class label, where each tree is grown in some randomized fashion. Each decision tree classifier consists of decision (or split) nodes and prediction (or leaf) nodes. The prediction nodes of each tree in the random forest classifier are labeled by the posterior distribution over the image classes [14]. Each decision node contains a test that splits best the space of data to be classified. An image is classified by sending it down the decision tree and aggregating the reached leaf posterior distributions. Randomness is usually injected at two points during training: when sub-sampling the training data and when selecting node tests. Each tree within the random forest classifier is binary and grown in a top-down manner. We choose the binary test at each node by maximizing the information gain,

$$\Delta E = -\sum_i \frac{|\, Q_i \,|}{|\, Q \,|} E(Q_i) \tag{2}$$

obtained by partitioning the training set Q of image patches into two sets Q_i according to a given test. Here $E(q)$ is the entropy of the set q and $|\cdot|$ is the size of the set. We repeat this selection process for each decision node until it reaches a

Authorized licensed use limited to: Sterne Kessler Goldstein Fox. Downloaded on November 05,2025 at 17:07:21 UTC from IEEE Xplore. Restrictions apply.

certain depth. Many implementations of random forests [18], [19] use simple pixel-level tests at the nodes because it results in faster tree convergence. As we are interested in features that encode the shape, and appearance, we are interested in spatial correspondence between the pixels. Therefore, we use a simple test proposed by [14] — a linear classifier on the feature vector — at each decision node.

Suppose T is the set of all trees, C is the set of all classes, and L is the set of all leaves for a given tree. During training, the posterior probabilities $(P_{t,l}(Y(I) = c))$ for each class $c \in C$ at each leaf node $l \in L$, are found for each tree $t \in T$. These probabilities are calculated as the ratio of the number of images I of class c that reach a leaf node l to the total number of images that reach that leaf node l. $Y(I)$ is the class label for image I. During test time, we pass a new image through every decision tree until it reaches a prediction (or leaf) node, average all the posterior probabilities, and classify the image as

$$\hat{Y}(I) = \arg\max_{c} \left\{ \frac{1}{\mid T \mid} \sum_{t=1}^{|T|} P_{t,l}(Y(I) = c) \right\} \tag{3}$$

where l is the leaf node reached by the image I in tree t. A graphical representation of the proposed random forest classifier is shown in Fig. 2 (see the right side).

III. Experiments and Results

A. Dataset

The Kaggle competition [3] for driver distraction has provided 22425 images for training and 79727 for test. Since we did not have access to the test labels, our experiments were done solely on the training images. However, the quality and conditions of training and testing images are similar; the only difference is that none of the drivers used in the training dataset appear in images in the test dataset. The images are of size 640×480, and for our experiments we converted them from color to grayscale.

There are ten classes provided, related to the ones listed in Section I. Each class includes almost tens of the data, so that we have a uniform distribution of sample data.

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

B. Algorithm Parameters

The convolutional neural random forest classifier is implemented using TensorFlow [20], and runs on an NVIDIA GeForce GTX TITAN X GPU with 16GB of memory. The classifier was trained using the stochastic gradient descent algorithm, Adam [13], to efficiently optimize the weights of the CNN. The weights were normalized using initialization as proposed in [13] and updated in a mini-batch scheme of 128 candidates. The biases were initialized with zero, and the learning rate was set to $\alpha = 0.001$. The exponential decay rates for the first- and second-moment estimates were set as $\beta_1 = 0.9$ and $\beta_2 = 0.99$, respectively. We used $\epsilon = 10^{-8}$ to prevent division by zero. A dropout rate of 0.5 was implemented as regularization, applied to the output of the last convolutional layer and the dense layer to avoid overfitting. Finally, we used an epoch size of 50. The softmax loss (cross-entropy error loss) was utilized to measure the error loss. We used 100 estimators and a keep rate of $\gamma = 10^{-4}$ for the random forest algorithm.

C. Performance Evaluation

We tested the algorithm performance by conducting a *k*-fold cross validation on the entire dataset. For our experiments we varied values of $k \in [2, 5]$ and found that the results were consistent enough to indicate that the network is not overfitting. Hence, we chose $k = 5$. First, we randomized the order of the driver images within the dataset. For each fold of the *k*-fold cross validation we chose 80% of the total 22425 images as the training dataset and tested the trained model on the remaining 20% of the images. We made sure that the images from the entire dataset appeared in the test dataset only once in all of the *k*-folds, thereby allowing each image to be classified as a test image exactly once.

We compared our proposed Drive-Net with two other neural network classifiers: a RNN classifier [21], and a MLP classifier [22]. We report the classification accuracy, which is defined as the percentage of correct predictions and the number of false positives (a.k.a. false detections) for each class as the figures of merit for comparing the algorithms. For classification accuracy, we present the results of seven other methods based on support vector machines (SVMs), dimensionality reduction techniques such as principal component analysis (PCA), feature extraction techniques such as histogram of oriented gradients (HOG), very deep convolutional nets such as VGG-16, VGG-GAP and an ensemble of these two as reported by Zhang in [17] using the same Kaggle dataset of 22425 images.

Table I shows the mean classification accuracy of the different classifiers as reported in Zhang [17] and that of the three neural network classifiers that we implemented. From Table I, we observe that the Drive-Net achieves a classification accuracy of 4.8 percentage points greater than the VGG-16 classifier, 3.7 percentage points greater than a VGG-GAP classifier, 2.4 percentage points greater than an ensemble of VGG-16 and VGG-GAP classifier, 3.3 percentage points greater than the RNN classifier and 13 percentage points greater than the MLP classifier.

Table II shows the number of false classifications for our Drive-Net and for the RNN and MLP. From Table II, we observe that our Drive-Net is able to identify the classes c6 (drinking) and c3 (texting with left hand) with minimum false

Authorized licensed use limited to: Sterne Kessler Goldstein Fox. Downloaded on November 05,2025 at 17:07:21 UTC from IEEE Xplore. Restrictions apply.

TABLE I
MEAN CLASSIFICATION ACCURACY OF THE AUTOMATED METHODS

Method	Accuracy
Methods from [17]:	
Pixel SVC	18.3%
SVC + HOG	28.2%
SVC + PCA	34.8%
SVC + Bbox + PCA	40.7%
VGG-16	90.2%
VGG-GAP	91.3%
Ensemble VGG-16 and VGG-GAP	92.6%
Methods we implemented:	
MLP	82 %
RNN	91.7%
Drive-Net	95 %

TABLE II
THE ERROR COUNT OF EACH CLASS FOR THE NEURAL NETWORK METHODS

Class	Drive-Net	MLP	RNN
c0	35	356	48
c1	17	199	34
c2	14	158	31
c3	09	116	47
c4	34	252	30
c5	15	108	14
c6	08	263	18
c7	21	117	10
c8	29	181	46
c9	26	268	46
All Classes	208	2018	324

detections, whereas the RNN and MLP classifiers have a hard time distinguishing these classes with many false detections, usually higher than the number of false detections in the other classes of these methods. Also, the total number of false detections for our Drive-Net is an order of magnitude smaller than that of the MLP classifier and slightly smaller in comparison to the RNN classifier.

IV. CONCLUSION

Distracted driving is a major cause of motor vehicle accidents. Therefore, there is a significant interest in finding automated methods to recognize signs of driver distraction from dashboard camera images installed in vehicles. We propose a solution to this problem using a supervised learning framework. Our method named Drive-Net combines a CNN and a random forest classifier to recognize the various driver distraction categories in images. We apply our Drive-Net to a publicly available dataset of images used in a Kaggle competition and show that our Drive-Net achieves better accuracy than the driver distraction algorithms reported in the competition. We also compared Drive-Net to two other neural network algorithms: a RNN and a MLP algorithm, using the same dataset. The results show that Drive-Net achieves better detection accuracy compared to the other two algorithms.

REFERENCES

[1] National Center for Statistics and Analysis. Distracted Driving: 2015, in Traffic Safety Research Notes. DOT HS 812 381. March 2017, National Highway Traffic Safety Administration: Washington, D.C.

[2] "Insurer Analyzes Top 10 Driving Distractions Involved in Fatal Car Crashes," *Insurance Journal*, 2013. [Online]. Available:https://safestart.com/news/top-10-causes-distracted-driving-and-what-they-all-have-common

[3] StateFarm, "State Farm Distracted Driver Detection," 2016. [Online]. Available: https://www.kaggle.com/c/state-farm-distracted-driver-detection

[4] A. Borji, M.-M. Cheng, H. Jiang, and J. Li, "Salient Object Detection: A Survey," *arXiv preprint arXiv:1411.5878*, 2014. [Online]. Available: https://arxiv.org/pdf/1411.5878.pdf

[5] R. C. Philip, S. Ram, X.Gao, and J. J. Rodríguez, "A comparison of tracking algorithm performance for objects in wide area imagery," *Proc. IEEE Southwest Symp. Image Analysis and Interpretation*, pp. 109–112, Apr. 2014.

[6] S. Ram, "Sparse representations and nonlinear image processing for inverse imaging solutions," Ph.D. dissertation, Dept. Elect. Comput. Engg., Univ. Arizona, Tucson, AZ, 2017.

[7] X.Gao, S. Ram, and J. J. Rodríguez, "A performance comparison of automatic detection schemes in wide-area aerial imagery," *Proc. IEEE Southwest Symp. Image Analysis and Interpretation*, pp. 125–128, Apr. 2016.

[8] S. Ram, and J. J. Rodríguez, "Vehicle detection in aerial images using multiscale structure enhancement and symmetry," *Proc. IEEE Intl. Conf. Image Process. (ICIP)*, pp. 3817–3821, Sep. 2016.

[9] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, inception-resnet and the impact of residual connections on learning," Presented at *Proc. of Thirty-First AAAI Conf. Artifical Intell.*, 2017. [Online]. Available: https://arxiv.org/pdf/1602.07261.pdf

[10] R. Girshick, J. Donahue, T. Darrell, and J. Malik. "Region-based convolutional networks for accurate object detection and segmentation." *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp.142–158, Jan. 2016.

[11] P.-K. Kim, and K.-T. Lim, "Vehicle type classification using bagging and convolutional neural network on multi view surveillance image," *Proc. of IEEE Conf. Comput. Vis. Pattern Recog. Workshop*, pp. 41–46, Jun. 2017.

[12] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: convolutional networks for biomedical image segmentation," *Proc. Med. Image Comput. Comput.-Assist. Intervention (MICCAI)*, pp. 234–241, Sep. 2015.

[13] D. Kingma, and J. Ba, "Adam: a method for stochastic optimization," Presented at *Proc. 3rd Intl. Conf. Learning Representations (ICLR)*, 2015. [Online]. Available: https://arxiv.org/pdf/1412.6980.pdf

[14] A.Bosch, A. Zisserman, and X. Munoz, "Image classification using random forests and ferns," *Proc. Intl. Conf. Comput. Vis.*, pp.1–8, Jul. 2007.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural netowrks," *Proc. Conf. Neural Information Processing Systems* pp. 1106–1114, Dec. 2012.

[16] K. Simonyan, and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Presented at *Proc. 3rd Intl. Conf. Learning Representations (ICLR)*, 2015. [Online]. Available: https://arxiv.org/pdf/1409.1556.pdf

[17] B. Zhang, "Apply and Compare Different Classical Image Classification Method: Detect Distracted Driver", Stanford CS 229 Project Reports, 2016. [Online]. Available: http://cs229.stanford.edu/proj2016/report/Zhang-DetectDistractedDriver-report.pdf

[18] J. Winn and A. Criminisi, "Object class recognition at a glance," *Proc. of IEEE Conf. Comput. Vis. Pattern Recog.*, pp.1082–1090, Jun. 2006.

[19] V. Lepetit, and P. Fua, "Keypoint recognition using randomized trees," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp.1465–1479, Sep. 2006.

[20] M. Abadi, A. Agarwal, P. Barham, *et al*, "TensorFlow: large-scale machine learning on heterogeneous systems," March 16, 2016. [Online]. Available: https://arxiv.org/pdf/1603.04467.pdf

[21] M. Liang, and X. Hu, "Recurrent convolutional neural network for object recognition," *Proc. of IEEE Conf. Comput. Vis. Pattern Recog.*, pp. 3367–3375, Jun. 2015.

[22] S. Haykin, *Neural Networks and Learning Machines,* Upper Saddle River, NY, USA:Pearson, 2009.

Authorized licensed use limited to: Sterne Kessler Goldstein Fox. Downloaded on November 05,2025 at 17:07:21 UTC from IEEE Xplore. Restrictions apply.