2018 IEEE 8th International Conference on Consumer Electronics - Berlin (ICCE-Berlin).

Recognition of driver distractions using deep learning

Leonel Cuevas Valeriano
Norwegian University of Science and Technology.
NTNU i Gjøvik
Norway
leonelcv@stud.ntnu.no

Paolo Napoletano, Raimondo Schettini
University of Milano-Bicocca
Department of Informatics, Systems and Communication
Viale Sarca 336, 20126 - Milano, Italy
{napoletano,schettini}@disco.unimib.it

***Abstract*—Driver distraction has a great impact on the safety of people and it is a relevant topic for a number of applications, from autonomous driving assistance to insurance companies and investigations. In this paper we address the problem of automatic recognition of driver distractions by exploiting deep learning and convolutional neural networks. We propose and present a comparison of different deep learning-based methods to classify driver's behaviour using data from 2D cameras. Evaluation has been carried out on the State Farm dataset, which consists of 10 different actions performed by 26 subjects such as, normal driving, texting, talking on the phone, operating the radio, drinking, reaching behind, etc. Results, achieved using 3 rounds of 5-fold cross validation, show that all the evaluated methods exceed the 90% of accuracy with the best achieving about 97%.**

***Index Terms*—Distracted driver, Action recognition, Deep learning, Convolutional Neural Network**

I. Introduction

The study of driver action recognition has been slowly gaining attention over the last decade due to its diverse number of applications, including those for improving the safety of drivers and passengers, providing driving assistance, supplying information to insurance companies and investigations and even for self-driving cars in situations when there might be a need for a human to take over control of the vehicle [1].

According to the data published by the National Highway Traffic Safety Administration (NHTSA), in 2015, in the USA, 391,000 people were injured in motor vehicle crashes involving distracted drivers, while 3,477 were killed [2]. Another of NHTSA's 2015 traffic safety report indicates that driver's misbehaviour is the critical reason for 94% of the car accidents, with mechanical problems or environmental circumstances being the critical reason for less than 5% [3]. Furthermore, it has been stated that 37% of the drivers admit to check and answer their texts, with 18% doing so regularly while operating a vehicle, and an alarming 86% of drivers report any of the following: eating, drinking, using their GPS system / checking a map, watching a video, surfing the web or grooming [4] [5].

These previous statements further motivate the need to identify a way to reduce the distraction of drivers on the road, as pointed out by [6]: *inattention while driving can increase the chance of getting into a motor vehicle crash and many of these accidents could have been prevented had the driver been warned the moment he got distracted*. Research by Dumitru earlier in 2018 demonstrated that by providing feedback when drivers are not focused on the road: on average, the number of driving infractions were reduced by 43.43%, the number of lane departures was reduced by 32.198%, the number of space cushions by 54.662% and the average speed decreased by 10.506% [7], so there is actually a clear positive reaction to providing feedback to drivers on their behaviour.

While this problem might seem to be fading away due to the introduction of self-driving vehicles, the fact is that the recent models of self-driving cars (as well as those used for providing private transport services) require their operator to remain attentive in case there is a need to take over [1]. According to Lyft's (a private transport company) FAQ site [8]: The pilot of a self-driving car might take over if there are any obstacles on the vehicle's route, such as constructions, traffic re-directions or a complex traffic situations, such as bad weather or unclear lane markings or signage. They also provide information on how the driver is notified and trained for being able to handle these situations successfully.

However nowadays, due to recent events, there are many doubts regarding the safety and reliability of these technologies, an example of this is the incident in March 2018, where a 49-year-old pedestrian was killed by one of Uber's self-driving vehicles [9], so in order to have more safety in these cases, being able to detect drivers distraction and notify it accurately and on time is a key factor.

In this paper we propose and present a comparison of different deep learning-based methods to classify driver's behaviour using data from 2D cameras. To the best of our knowledge, most of previous studies focus only on frame-based classification methods while in this paper we evaluate video classification methods. For the task of driver distraction detection, the use of hand-crafted features has demonstrated to perform worse than deep-learning-based features combined with a Support Vector Machine (SVM) with results up to three times worse [10].

This paper is organized as follows: in section II we present a literature review with previous studies and their approaches, in section III we present our methodology, in section IV we discuss and analyze the results obtained and, finally, in section

978-1-5386-6095-9/18/$31.00 ©2018 IEEE

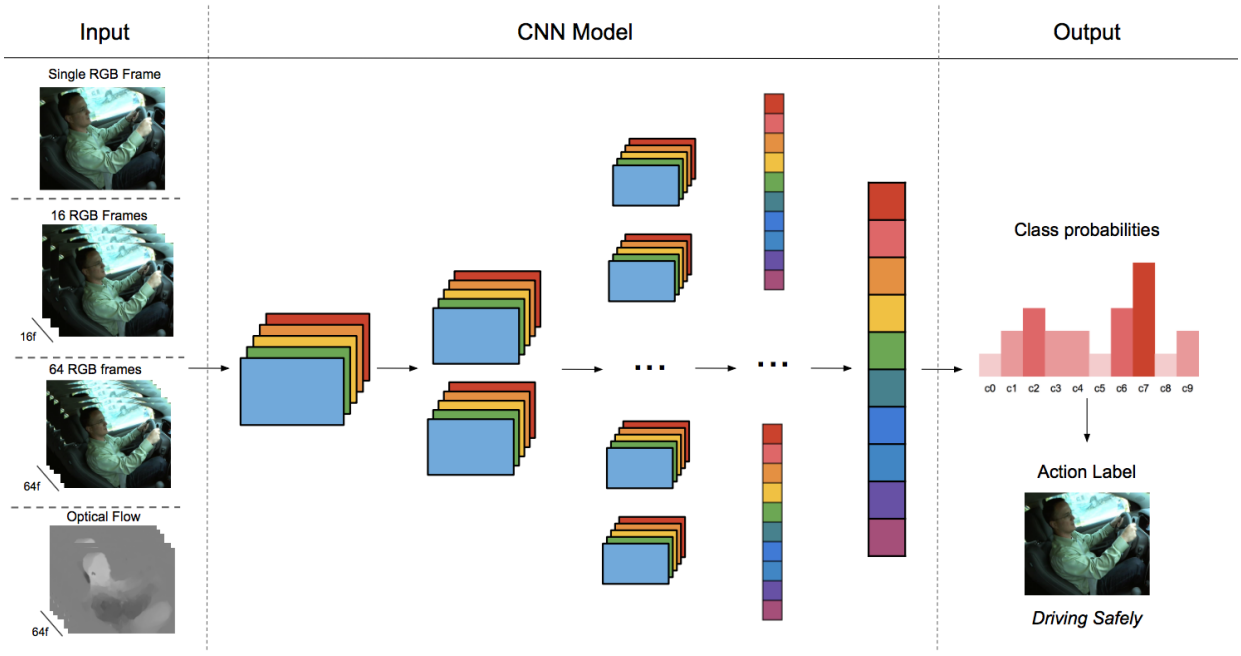

Fig. 1. Schematic showing the overall process of our experiments

V we take a look at the final considerations of the work done and present our conclusions.

II. Literature Review

Previous research has been conducted for studying deep leaning methods for the problem of driver distraction detection.

In [11] Yan *et al.* present a Convolutional Neural Network (CNN) trained by using pre-trained sparse filters [12] as the parameters of the first convolutional layer, after that they proceed to do the fine-tuning of the CNN using their dataset.

In [13] Abouelnaga *et al.* propose an ensemble of 10 CNNs (5 AlexNet [14] and 5 InceptionV3 [15]), where the "voting system" for this ensemble is determined by a Genetic Algorithm and where different CNNs are trained on different processed data such as the original non-processed frame, face-segmented frame, hands-segmented frame, face+hands-segmented frame or a skin-segmented frame. They showed that the segmentation steps proved to improve classification accuracy, but that, in a real-time setting, their performance overhead is too high to be considered.

In [10] Hssayeni *et al.* explored the use of AlexNet [14], VGG-16 [16] and ResNet-152 [17] architectures and showed that the more modern and deeper architectures perform better, they also compared the use of CNNs for classification vs. CNNs for feature extraction + SVM for classification, showing that the latter method did not increase the accuracy.

More recently, Masood et al. [6] evaluated the use of pretrained VGG-16 and VGG-19 [16] models and data augmentation and proved the use of these techniques to be very effective for improving their classification results while also reducing the training time.

Other research studies exist that are based on tracking the driver's gaze and attention using head position [18], segmenting and tracking hands through the use of depth images and chroma-keying gloves [19], and even more complex setups which make use of electroencephalograms signals on the forehead to detect fatigue and alertness [20], [21].

III. Materials and Method

In this work we explore four different approaches: the first is a single-frame based classification, similar to [10]; the rest are video-based methods with different input types being: 16 frames, 64 frames and Optical Flow.

A. Dataset

For training and testing we make use of State Farm dataset [22], which was released on 2016 by State Farm Insurance Co. as part of a Kaggle competition for image-based driver posture classification. The dataset consists of 22,424 labeled images, from 26 subjects (of different age, ethnic group, sex, size, skin color, etc.). The subjects were instructed to perform 10 different actions (normal/safe driving, texting with right hand, talking on the phone with right hand, texting with left hand, talking on the phone with left hand, operating the radio, drinking, reaching behind, hair and makeup and talking to passenger), each image is labeled with the appropriate action class.

The dataset is aimed to be used for frame-based classification and the samples are labeled on a frame by frame basis. It is possible to reconstruct the original video sequence and use it as input. In this case, the dataset is structured as 260 videos with varying duration (10 for each action for each subject).

B. Methodology

All the strategies evaluated are quite similar with some slight variations either on the architecture of the CNNs or the input used, so we describe the general method and later focus on the differences for each particular method. The general process is shown in Figure 1.

Due to the small amount of data available, it is unfeasible to train a CNN from the scratch. For this reason, we fine-tune pretrained models: in the case of the frame based method we use a model pretrained on the ImageNet dataset [23], while for the video-based methods we use models pretrained on the Kinetics dataset [24]. Additionally we performed several data augmentation techniques to diversify the dataset, these include, random scale, random crops, random rotation, random temporal crops, etc.

All the strategies are implemented using the Pytorch [25] deep learning framework. We fine-tune the last layers of the pretrained models and perform 3 rounds of 5-fold cross-validation on each of them. For each case 18 subjects (180 videos) were used for the training set, and 6 subjects (60 videos) for test, in the image-based experiments, the corresponding frames of the videos assigned are used for training and test. For every run, each of the 26 subjects only appears in either training or validation sets.

It is important to note that, for the video-based methods, we decided to use 3D CNN architectures since they can model appearance and motion information simultaneously and due to their better performance when compared to handcrafted features and LSTM-based methods [26].

1) Single Frame Input: For this method we make use of the pretrained ResNet-101 model [17] available in the Pytorch's torchvision module [27]. This model has selected after comparing it with different pretrained models.

2) 16 Frames Input: In this case we used a ResNext-101 [28] pretrained model with 16 consecutive RGB frames of video data as input. More specifically, we used the implementation by Kensho Hara [29], we selected this after comparing how it performed with respect other models (such as the RGB branch of I3D model [30], [31]).

3) 64 Frames Input: In this case we also used a ResNext-101 [28] pretrained model with 64 consecutive RGB frames of video data as input. More specifically, we used the implementation by Kensho Hara [29], again we selected this after comparing its results with other models.

4) Optical Flow Input: In this case we used the optical flow branch of the I3D [30] pretrained model, more specifically, the implementation found in [32]. The I3D model [30] is one of the state-of-the-art methods for action recognition. As input 64 frames of dense optical flow obtained from consecutive frames are given to the CNN.

The idea behind the use of optical flow is that it explicitly represents and models the motion in the video, which is expected to make the process simpler, since the CNN does not need to estimate motion [33]. We calculated the optical flow using the TV-L1 algorithm [34], just like in the original work.

TABLE I
RESULTS OF THE EXPERIMENTS PERFORMED

Method (Input Type)	Average Accuracy (%)	Standard Deviation	Performance (FPS)
Single Frame	82.38	7.31	29.767
16 Frames	84.41	4.16	24.833
64 Frames	91.25	1.28	14.430
Optical Flow	92.74	1.81	<1

IV. EXPERIMENTS

A. Results and Analysis

We performed 3 rounds of 5-fold cross validation. A summary of the results for the testing accuracy of each method is presented in Table I. Also in figure 2 we can see examples of the confusion matrix obtained for every case.

From this results we can observe that in general video-based methods perform better than the frame-based ones, we can attribute this to the additional information encoded into the video input itself. In addition to this, in the confusion matrix for the single frame methods in figure 2 we can observe that there is a big misclassification of passenger interaction frames with normal driving. This kind of error is attributed to the lack of temporal information in the input, some frames in the dataset can be very ambiguous for those classes, since even while performing a specific action the drivers tend to look back to the road, which might be interpreted as normal driving.

On the other hand, while the results improvement for the video-based methods significantly lowering the confusion between the normal driving and passenger interaction classes (which we attribute to the temporal information that comes with this kind of input), we can see that some new misclassifications are introduced such as the one observed in the 16 Frames confusion matrix of figure 2 where call right and drink actions are misclassifies. This could be attributed to the fact that the 3D CNNs "learn" descriptors based on appearance and motion, so when two actions share some similarities in their motion (such as in this case rising the right hand to he head-level) there is the risk for misclassification if the features extracted are not robust enough. We see an improvement on this when more data is supplied in the 64 frames and optical flow input experiments.

The optical flow model was the best performing showing the best average accuracy, however its results are not so far from those obtained by using 64 RGB frames as input.

By looking at the standard deviation is very noticeable that for the single frame method the value is one order of magnitude higher for the video-based methods, indicating that its performance might be more reliant on the selection of the train/test split, which makes sense due to the presence of ambiguous frames in the dataset. On the other hand video-based methods showed more stable results, however this comes, of course at the cost of more computing and training time.

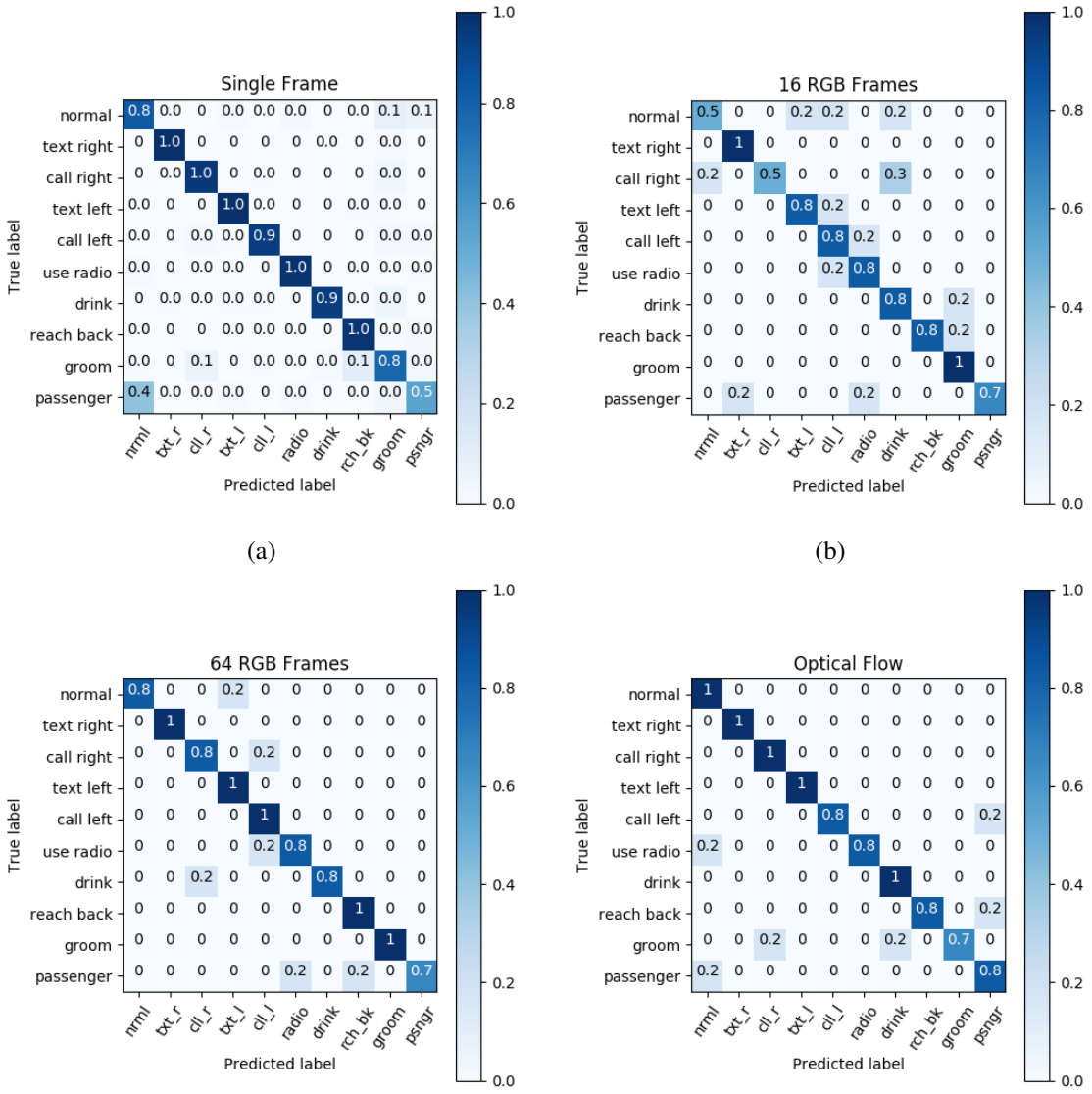

Fig. 2. Examples of Confusion matrices obtained for each method: a) single frame input, b) 16 RGB frames input, c) 64 RGB frames input, d) optical flow input

One thing we consider is important to note is the difficulty to compare single frame with video based methods, for the latter the number of samples is significantly lower, so a single failure in video dataset brings the results significantly lower compared to one misclassification on the image-based scenario. Even when comparing the video-based methods this could be an issue, since due to the weight of each sample being so high, in the end their results might be closer than they appear and the performance might actually be similar across all video-based methods; but in any case, we consider that the results obtained are illustrative of the potential for each methods as well as for their shortcomings.

As a final experiment, we decided to try combining both RGB and optical flow using the I3D architecture, just as they do in [30]. For this experiment we didn't need to perform any training instead simply joined our pretrained models with best performance and apply it on the test set; as mentioned in section III-B2 we also trained on the RGB branch of I3D model, we just do not report the results here since it was outperformed by ResNext-101. The confusion matrix can be seen in figure 3 and as it can be observed the accuracy improved significantly obtaining an average of 96.67% across all different classes, which proves the effectiveness of this hybrid approach. The method, however still suffers from the same shortcomings as the ones mentioned for the 64 RGB frames and Optical Flow methods, plus an additional hit on performance and resources, due to the need of running both models in parallel, but the improvement in accuracy might be worth this in some applications.

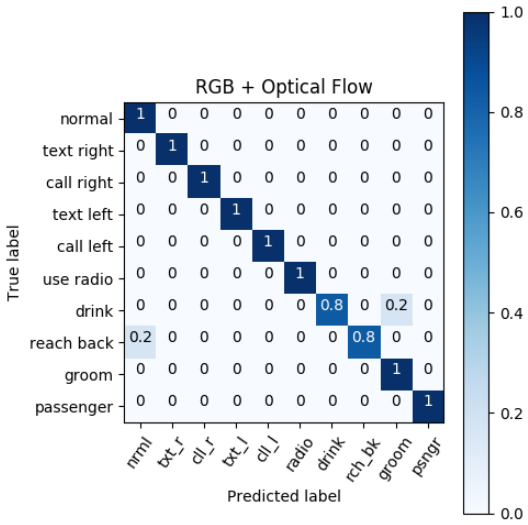

Fig. 3. Confusion matrix obtained by combining the RGB frames with the optical flow.

V. Conclusion

In this paper we presented four different approaches to solve the driver distraction detection problem, with our best model, which makes use of both RGB and optical flow input, achieving a 96.67% of classification accuracy. These results also prove the effectiveness of transfer learning, since even despite the low amount of data, it was possible to obtain good results which prove the feasibility of this task; eventually, if required, we consider that the system can be improved and refined even further with more data.

One of the potential issues with this kind of system is related to the privacy of the drivers, there is a possibility that a number users would not feel comfortable having a camera monitoring all their actions, however the benefits and impact of systems which can provide feedback to the drivers has already been proved [7] and this could out-weight the negative aspects.

As for future work we would like to try using skeleton images as input, we consider this could help to reduce the influence of the background environment on the results; or also introduce the use of other sensor data that could help us to get better results or to improve the discrimination capabilities of the system and introduce more action classes.

Acknowledgment

This research has been developed in the context of the projects: 1) TEINVEIN, TEcnologie INnovative per i VEicoli Intelligenti, CUP (Codice Unico Progetto - Unique Project Code): E96D17000110009 - Call "Accordi per la Ricerca e l'Innovazione", cofunded by POR FESR 2014-2020 (Programma Operativo Regionale, Fondo Europeo di Sviluppo Regionale – Regional Operational Programme, European Regional Development Fund); 2) The Home of Internet of Things (Home IoT), CUP: E47H16001380009 - Call "Linea R&S per Aggregazioni" cofunded by POR FESR 2014-2020.

References

[1] A. Eriksson and N. A. Stanton, "Takeover time in highly automated vehicles: noncritical transitions to and from manual control," *Human factors*, vol. 59, no. 4, pp. 689–705, 2017.

[2] NHTSA, "Distracted driving," https://www.nhtsa.gov/risky-driving/distracted-driving/, 2015, accessed: 2018-05-17.

[3] ——, "Traffic safety facts: Crash stats," 2015.

[4] E. Ohn-Bar, S. Martin, A. Tawari, and M. M. Trivedi, "Head, eye, and hand patterns for driver activity recognition," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*. IEEE, 2014, pp. 660–665.

[5] S. Duan, T. Yu, and J. He, "Widriver: Driver activity recognition system based on wifi csi," *International Journal of Wireless Information Networks*, vol. 25, no. 2, pp. 146–156, 2018.

[6] S. Masood, A. Rai, A. Aggarwal, M. Doja, and M. Ahmad, "Detecting distraction of drivers using convolutional neural network," *Pattern Recognition Letters*, 2018.

[7] A. I. Dumitru, T. Girbacia, R. G. Boboc, C.-C. Postelnicu, and G.-L. Mogan, "Effects of smartphone based advanced driver assistance system on distracted driving behavior: A simulator study," *Computers in Human Behavior*, 2018.

[8] Lyft, "Most frequently asked questions about self-driving experience," https://www.lyft.com/self-driving-vehicles/faq, 2018, accessed: 2018-05-21.

[9] D. Wakabayashi, "Self-driving uber car kills pedestrian in arizona, where robots roam," *The New Yok Times*, 2018. [Online]. Available: https://www.nytimes.com/2018/03/19/technology/uber-driverless-fatality.html

[10] M. D. Hssayeni, S. Saxena, R. Ptucha, and A. Savakis, "Distracted driver detection: Deep learning vs handcrafted features," *Electronic Imaging*, vol. 2017, no. 10, pp. 20–26, 2017.

[11] C. Yan, F. Coenen, and B. Zhang, "Driving posture recognition by convolutional neural networks," *IET Computer Vision*, vol. 10, no. 2, pp. 103–114, 2016.

[12] J. Ngiam, Z. Chen, S. A. Bhaskar, P. W. Koh, and A. Y. Ng, "Sparse filtering," in *Advances in neural information processing systems*, 2011, pp. 1125–1133.

[13] Y. Abouelnaga, H. M. Eraqi, and M. N. Moustafa, "Real-time distracted driver posture classification," *arXiv preprint arXiv:1706.09498*, 2017.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[17] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[18] Y. Yun, I. Y. Gu, M. Bolbat, and Z. H. Khan, "Video-based detection and analysis of driver distraction and inattention," in *Signal Processing and Integrated Networks (SPIN), 2014 International Conference on*. IEEE, 2014, pp. 190–195.

[19] A. Rangesh and M. M. Trivedi, "Handynet: A one-stop solution to detect, segment, localize & analyze driver hands," *arXiv preprint arXiv:1804.07834*, 2018.

[20] Z. Mu, J. Hu, and J. Yin, "Driving fatigue detecting based on eeg signals of forehead area," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 31, no. 05, p. 1750011, 2017.

[21] P. Napoletano and S. Rossi, "Combining heart and breathing rate for car driver stress recognition," in *International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*. IEEE, 2018, pp. –.

[22] S. Farm, "State farm distracted driver detection - data," https://www.kaggle.com/c/state-farm-distracted-driver-detection/data, 2016, accessed: 2018-05-21.

[23] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, pp. 248–255.

[24] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev *et al.*, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[25] J. Reed, "Pytorch," https://github.com/pytorch/pytorch, 2017.
[26] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *Computer Vision (ICCV), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4489–4497.
[27] PyTorch, "Pytorch vision," https://github.com/pytorch/vision, 2017.
[28] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*. IEEE, 2017, pp. 5987–5995.
[29] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet?" *arXiv preprint*, vol. arXiv:1711.09577, 2017.
[30] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 4724–4733.
[31] G. Ciocca, A. Elmi, P. Napoletano, and R. Schettini, "Activity monitoring from rgb input for indoor action recognition systems," in *International Conference on Consumer Electronics - Berlin (ICCE-Berlin)*. IEEE, 2018, pp. –.
[32] Hassony, "I3d models transfered from tensorflow to pytorch," https://github.com/hassony2/kinetics_i3d_pytorch, 2017.
[33] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in neural information processing systems*, 2014, pp. 568–576.
[34] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime tv-l 1 optical flow," in *Joint Pattern Recognition Symposium*. Springer, 2007, pp. 214–223.