Distracted Driver Detection with Convolutional Neural Network

Shijie Liu
Computer Science Department
Hood College
Frederick, Maryland, USA
sl23@hood.edu

ABSTRACT

Safety has become more and more important with the rapidly development of highway infrastructure around the world. Organizations like the World Health Organization (WHO) and National Highway Traffic Safety Administration (NHTSA) are collecting accident data statics yearly and performing analysis to understand the major causes. The analysis shows distracted driving is one of the major reasons. The Centers Disease Control and Prevention (CDC) Transportation Safety also defined three types of distraction: eyes off the road as visual distraction, hands off the wheel as manual distraction, and minds off driving as cognitive distraction [1]. State Farm, an insurance company had collected a dataset of 2D dashboard camera images that are properly labeled with 9 distraction classes identified, and the dataset is available on Kaggle for competition [2]. Therefore, this project aims to build and train convolutional neural network models that can detect and classify distracted drivers, which includes a convnet from scratch and a fine-tuned VGG-19 network. Throughout this project, techniques like image augmentation, batch normalization, and dropout are used to overcome the overfitting issue. At end of the project, the best model can achieve 87% prediction accuracy with a small labeled dataset that the model never seen before.

CCS CONCEPTS

• Computing methodologies → Artificial intelligence; Computer vision; Computer vision tasks; Activity recognition and understanding; Artificial intelligence; Computer vision; Computer vision tasks

KEYWORDS

Deep Learning, Computer Vision, Convolutional Neural Network, Transfer Learning, Feature Extraction, Fine tuning, Deep Neural Network, Image Augmentation, Batch Normalization, Distracted Driver

1 INTRODUCTION

With the assessment of road safety in 178 countries, the World Health Organization (WHO) had reported there are approximately 1.3 million people die each year on the world's roads, and between 20 and 50 million sustain non-fatal injuries [9]. According to National Highway Traffic Safety Administration (NHTSA), in 2018, over 2,800 people were killed and an estimated 400,000 were injured in crashes involving a distracted driver in the U.S [3]. The Centers Disease Control and Prevention (CDC) Transportation Safety also defined three types of distraction: eyes off the road as visual distraction, hands off the wheel as manual distraction, and minds off driving as cognitive distraction. [1] State Farm, an insurance company had collected a dataset of 2D dashboard camera images that are properly labeled with 9 distraction classes identified, and the dataset is available on Kaggle for competition to classify driver's behavior [2].

This project aims to build and train convolutional neural network models that can classify different distraction classes, convolution layers can extract features, pooling layers can downsize the dimension of feature map by, and dense layers can get the probability distribution. The project starts with Data exploration and pre-processing. Then, two network architectures will be created with various combinations of different layers. Then, both networks will be trained with same set of data generators with same optimizer and loss function, and parameters will be adjusted as experiments. After models are trained, they will be used to perform predictions on a test dataset that is 100% inaccessible during model training phase. Afterwards, metrics like accuracy, loss, and confusion matrix will be used to compare the model performance.

2 BACKGROUND

In the United States, driving is a very usual thing for individuals. There are laws, such as no texting, no cellphone usage, and no make-up while driving, but not all the people obey the laws. I still remember a video from driver education class, which shows teenager drivers lost their life in car accidents, and the last thing they did was texting. Two years ago, I also heard a sad story from Dr. Lin, Pengxin, one of his employee's mom passed away in a car accident because her car was hit by a distracted truck driver.

With the Deep Learning class, the computer vision chapter attracts me a lot, especially the great benefit of Convolutional Neural Network and Transfer learning based on pre-trained models. Quoted from Dr. Liu, "if you provide data and tell the network how many classes to clarify, the network will learn on its own", which sounds fantastic to me. I strongly want to see if a Convolutional Neural Network can learn from labeled 2D dash-cam images by itself, classify driver actions and detect distracted drivers.

3 RELATED WORK

In the past years, the distracted driver issue has been studied by a lot people due to the alarming statics being reported every year and

more and more labeled data are becoming available, and the computing platforms are becoming more capable. Abouelnaga et al. [10] mentioned a cell phone detection study that was done by Berri and Silva [12], they used an SVM-based model for detecting cell phone usage, which uses frontal images of a driver's face, but hand and face location in the picture are the limitations of this model. Later, a Faster-RCNN model was devised by Hoang et al. [13], which can be used to detect whether driver is using cellphone or driver's hands are not on the wheel. Zhao et al. [14] proposed an efficient feature extraction approach for driving postures dataset, they used Random Forest (RF) classifier that was able to classify four actions: grasping the steering wheel, operating shift gear, eating, and talking on cellphone. With 88% accuracy, RF classifier outperforms when comparing with other classifiers like linear perceptron, k-nearest-neighbor, and multi-layer perceptron. Eraqi et al. [16] also collected and labeled their own dataset for their study. They proposed a solution that consists of a genetically weighted ensemble of convolutional neural networks that uses transferring learning and were trained on raw images, skin-segmented images, face images, hands images, face+hands images, the final distribution is based on a weighted sum of all networks' output.

With the distracted driver dataset provided by State Farm [2], Tamas and Maties [17] attempted to develop a real-time system based on CNN that is able to detect and identify the cause of distraction, which uses base architecture of VGG-16 with modifications, mainly on the activation functions. Another study is done by Mofid et al. [15], they utilized a combination of pertained image classification models, classical data augmentation, OpenCV based image preprocessing and skin segmentation augmentation approaches in order to overcome the limitation of limited number of drivers in the dataset. Omerustaoglu et al.[6] also studied a two stage system to detect distracted behavior. The first stage is a vision based Convolutional Neural Network that was created by transfer learning and fine-tuning methods, and the second stage used Long-Short Term Memory-Recurrent Neural Network (LSTM-RNN) that was created using sensor and image data together. They had concluded the accuracy of vision-based model increases with fine-tuning on pre-trained CNN model.

4 IMPLEMENTATIONS

4.1 Dataset Description

State Farm Distracted Driver Detection dataset [2] has been chosen for this project. The dataset is only available for competition that all image metadata (creation dates) has been removed. Drivers included in this dataset have been separated to train and test data and one driver can only appear in one set of data, which ensures the test data are 100% unknown to the trained model when performing prediction. All image data are collected from 2D dashboard cameras that captures different distraction activities of the driver, such as texting, talking on the phone, drinking, and reaching behind, etc. Below is a list of classes this dataset supports:

- 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

This dataset also includes a CSV file that captures image name, subject id, and class id for training data, which helps to explore the data distribution to ensure data balance before feeding the network.

4.2 Dataset Exploration

Exploratory data analysis (EDA) technique is being used for better understanding of the dataset, which involves several graphical techniques. For example, Bar Plotting is used for understanding the data distribution, Figure 2 shows number of images per class and Figure 3 shows number of images per driver.

Image reading and plotting are utilized for visualizing the image data, this technique could visualize a group of sample data that represents each distraction class (Figure 1)

4.3 Dataset Split

Based on the dataset exploration, there are 22424 images under the train directory, and this is the only directory that is properly labeled. I decided to split this train directory to 20% for validation and 80% for training, which gives us 17943 training images and 4481 validation images. To be realistic and accurate on model performance evaluation, relying on the train/validation loss and accuracy metrics are not enough since overfitting is a common recognized challenge for neural networks. Thus, I manually labeled 100 images to 10 classes from test directory to get a small test dataset that the trained model has zero knowledge on it, which will be used to check the prediction accuracy as part of our model performance evaluation.

Figure 1 Sample images for distraction classes

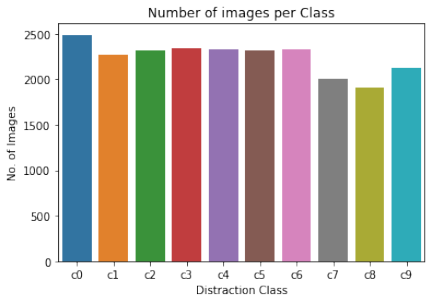

Figure 2 Number of images per distraction class

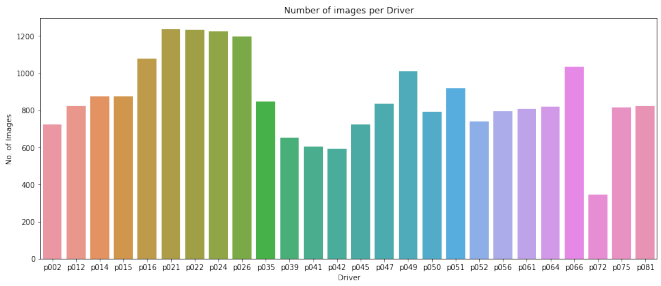

Figure 3 Number of images per driver

4.4 Dataset Pre-Processing

Images are formed by pixels with the value in range [0, 255], no matter grayscale image or colorful image. Rescaling 1. /255 has been applied as pre-processing to normalize the value to range [0, 1], which helps the neural network to have a higher chance converging since it makes more evenly total loss [18] and works well with a typical learning rate.

Image resizing has also been applied to reduce the memory load and offer a chance to increase the batch size. The original aspect ratio and spatial information of an image will still be kept, which will not affect the learning by a neural network. In this project, the visual difference for different image sizes have been experimented for determining a target size for the input of our neural networks. Among 256x256, 128x128, 64x64, 32x32, and 16x16, 64x64 has been chosen for this project with the consideration of the limited computing platform. (Figure 4)

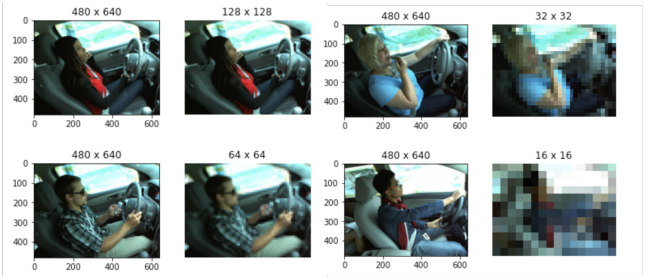

Figure 4 Image resizing

4.5 Architectures

As mentioned in the Related work section, people had taken different approaches for the distracted driver detection. Under the inspiration of the Computer Vision chapter from Francois's Deep learning with Python [4], I decided to build two neural networks: convolution neural network from scratch and fine-tuned VGG-19.

4.5.1 Convolution Neural Network from Scratch.

Although there are numerous variants among Convolution Neural Networks, convolution, pooling, and fully connected layers are commonly being used. Thus, this project also uses the common layers for the scratch convolution neural network. The input layer accepts 64x64x3 input because 64x64 is our target image size during pre-processing and there are 3 channels since our image data are loaded as RGB format. There are three convolution layers-The first convolution layer has 32 filters, the second convolution layer has 64 filters, and the third convolution layer has 128 filters. Each convolutional layer uses same *relu* activation function and 5x5 stride; following each convolutional layer is a max pooling layer that uses a 2x2 pooling window. Then, a flatten layer and dropout layer have been added, and another two fully connected dense layers follows. The last dense layer uses *softmax* activation function with an output size of 10. Typically, the last fully connected layer has the same number of output nodes as the number of classes to be classified. The architecture is also shown in Figure 5.

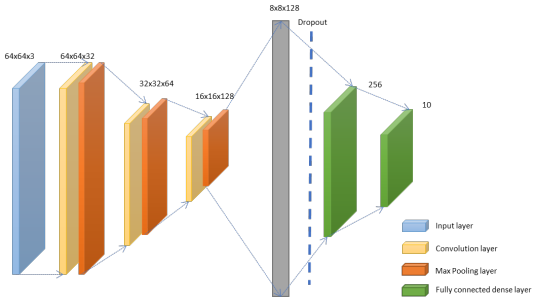

Figure 5 Architecture of scratch convolution neural network

4.5.2 Fine-tuned VGG-19.

The VGG network architecture was introduced by Simonyan and Zisserman in their 2014 paper, Very Deep Convolutional Networks for Large Scale Image Recognition [19]. The VGG-19 architecture accepts 224x224x3 as input, which will be updated to 64x64x3 for this project. It includes 16 convolutional layers, 5 max pooling layers, and 3 fully connected dense layers. As part of fine-tuning, the 3 fully connected dense layers will be updated to 2, with the output size of 256 and 10. Besides, a dropout also get added before the last fully connected layer. The modified VGG-19 architecthture is shown in Figure 6.

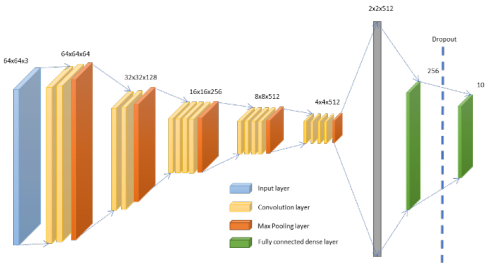

Figure 6 Architecture of fine-tuned VGG-19

4.6 Techniques

4.6.1 Image Data Generator.

The tradition way of loading a image dataset is load all data into memory via processing all files and create Numpy array for each image file, this approach has become a challenge for computer vision problems because of limited memory. For example, NVIDIA Jetson Nano[24] has a 128-core Maxwell GPU, but it runs very slow when training model. This project uses *ImageDataGenerator* provided by Keras[25] for dataset loading, it provides enough flexibility for loading data, such as categorizing class based on folder name, loading data via batching, splitting data into train and validation, applying image augmentation on the fly and image rescaling, resizing. However, it has limitations that it require the data to be labeled properly and correctly categorized to sub-folders.

4.6.2 Transfer Learning.

Using pre-trained network has been recognized as a common and highly effective approach for deep learning on a small image dataset, because those pre-trained models were previously trained on large dataset that spatial features for generic objects have been learned. For example, ImageNet dataset [26] includes 1.4 million labeled images for 1000 different classes. Most of the time, people who want to implement deep learning model do not have a large-scale image dataset because it takes time to collect, clean, and label. The State Farm distracted driver dataset [2] also being considered as a small image dataset since it only has 22424 labeled images for 10 classes and 26 drivers in total. Besides, Keras offers a wide range of deep learning models that are available with pre-trained weights, such as Xception, VGG-16, VGG-19, and RestNet50, etc. Thus, a pre-trained network has been chosen for this project.

There are two different approaches for transfer learning, feature extraction and fine-tuning. feature extraction removes fully connected layers from a pre-trained model while remains the reset series of convolutional and pooling layers, which serves as a convolutional base. Then, new classifier and fully connected layer will be added on the top, which result at a model that relies on the weights from pre-trained model. Fine-tuning replaces fully connected layers from pre-trained model with new set of fully connected layers and fine-tune all or part of kernels in the convolutional base by means of backpropagation. It offers the flexibility whether to fine-tune all layers or keep earlier layers frozen while fine-tuning the rest of layers [30]. Fine-tuning approach has been chosen for this project.

4.6.3 Overfitting.

Overfitting, which is observed when a model performs very well on training data but extremely poorly on unknown data [31]. This is so true that the training and validation accuracy are 95% plus, but the model gets much less accuracy while performing predictions with test dataset. One of the major causes is lack of data that the model starts to memorize what he had seen, or the dataset is unbalanced. If the dataset is infinite, the model would need to learn every possible aspect that it would never overfit. Salman and Lin also concluded that overfitting is due to continuous gradient updating and scale sensibleness of cross entropy loss [28]. Based on the identified causes, this project utilizes the following three techniques to overcome this issue:

4.6.3.1 Image Data Augmentation.

Data augmentation is just an approach of generating more data from existing data, which deals with the root cause of overfitting and offers a way of transforming available data to new data without altering its nature. Kostrikov also stated that Image augmentation techniques have been successfully used to overcome overfitting problems in computer vision related tasks [35].

There are two major approaches for Image data augmentation, traditional transformation and deep learning approach. The traditional approach includes image shifting, cropping, flipping, shearing and rotating [33]. Adversarial training, neural style transfer, and GAN (Generative adversarial networks) Data augmentation are examples of deep learning approach [20]. Only traditional transformation approach will be used in this project due to limited time, and *ImageDataGenerator* from Keras [25] supports transformation such like rotation, shifting, shearing, brightness, and zooming. It augments the image data on the fly, during training time. While exploring the dataset, I noticed some images became hard to recognize from human perspective, however, I still found data augmentation helps to improve the model performance in the end. That matches what Wang stated, “algorithms can perform better even when the data is of lower quality, if useful data can be extracted by the model [38].” Below is a plot of augmented images that shows a sample image being transformed in different ways (Figure 7).

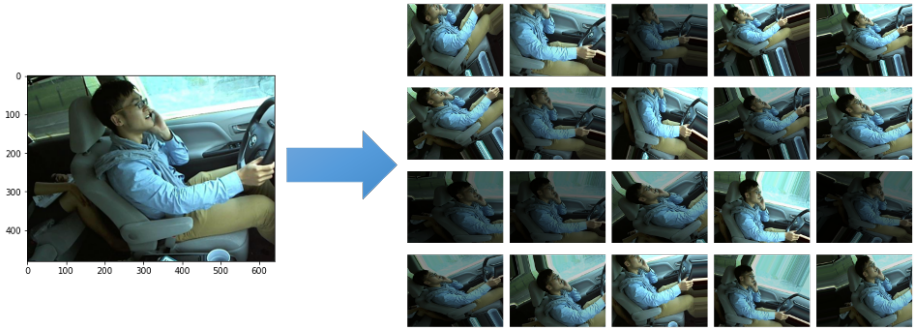

Figure 7 Image augmentation with *ImageDataGenerator*

4.6.3.2 Batch Normalization.

Although data normalization has been applied during data pre-processing, the distribution of input to internal layers still change when the data flows through the deep network. It will lose the learning capacity and accuracy of the network. According to Ioff et al. [8], batch normalization improves the performance and stability of neural networks by explicitly forcing the activations through a layer to follow a unit Gaussian distribution. Thus, batch normalization has been applied to all convolutional layers and fully connected dense layers for the neural network from scratch.

4.6.3.3 Dropout.

Dropout is another effective method for overfitting, it normally uses regularization techniques for neural networks, which is an efficient way of reducing overfitting by randomly drooping out some neurons during training phase [7].

4.7 Model Performance Evaluation

The computing platform is 2019 MacBook Pro that uses 1.4 GHz Intel Core i5 processor and 8GB memory, but it does not have GPU support. For experiment purpose, there are several variables are being adjusted for comparing the model performance. Such as dropout, data augmentation, and learning Rate, etc. Primarily, there are two groups of comparisons, how model behaves before and after changing a variable, and how two different model performs with the same input. The metrics are being used for evaluation are loss, accuracy, and confusion matrix.

Throughout the model training and evaluating, several parameters are manipulated and will be included in the tables below. Before proceeding, the following acronyms have been defined as, D represents Dropout, BN represents Batch Normalization, DA represents Data augmentation, LR represents Learning Rate, and LF represents Layers Frozen.

Table 1 shows how the scratch network changing performance with manipulated parameters, while Table 2 shows how the fine-tuned VGG-19 network changing performance with manipulated parameters.

4.7.1 Chosen scratch neural network

The chosen network for scratch version had 0.3 dropout rate, batch normalization, data augmentation, and 0.01 learning rate. Regarding the training and testing of the best scratch neural network, Figure 8 shows the loss, Figure 9 shows the accuracy, and Figure 10 shows the confusion matrix.

4.7.2 Chosen Fine-tuned VGG-19 network

The chosen fine-tuned VGG-19 network has 0.3 dropout rate, data augmentation, 0.0001 learning rate, and first 10 layers are frozen. Regarding the training and testing of the best scratch neural network, Figure 11 shows the loss, Figure 12 shows the accuracy, and Figure 13 shows the confusion matrix.

	Training		Validation		Test	
	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy
Base	0.022	99.33%	0.085	97.83%	2.69	66.00%
Base + D(0.3)	0.009	99.72%	0.022	99.46%	2.85	62.99%
Base + D(0.3) + BN	0.051	99.81%	0.041	99.62%	0.903	75.00%
Base + D(0.3) + BN + DA	0.207	94.98%	0.422	87.03%	0.664	81.00%
Base + D(0.2) + BN + DA	0.148	96.50%	0.221	94.11%	0.858	75.00%
Base + D(0.4) + BN + DA	0.343	91.52%	0.487	85.69%	0.762	74.00%
Base + D(0.3) + BN + DA + LR(0.01)	0.087	97.55%	0.074	97.88%	0.683	84.00%
Base + D(0.3) + BN + DA + LR(0.001)	0.199	95.12%	0.32	90.02%	0.804	75.00%
Base + D(0.3) + BN + DA + LR(0.0001)	0.59	88.65%	0.55	89.49%	0.96	70.00%
Base + BN + DA + LR(0.01)	0.0626	98.32%	0.078	97.63%	0.792	78.00%

Table 1 Scratch Network Performance

Figure 8 Loss

Figure 9 Accuracy

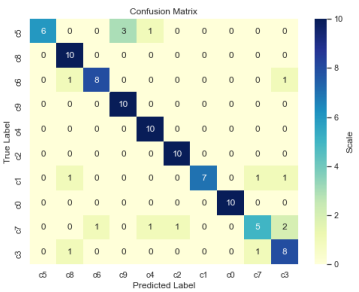

Figure 10 Confusion Matrix

	Training		Validation		Test	
	Loss	Accuracy	Loss	Accuracy	Loss	Accuracy
Base	0.013	99.88%	0.083	97.46%	2.42	54.00%
Base + D(0.3)	0.092	97.11%	0.071	98.12%	2.11	46.99%
Base + D(0.3) + DA	1.23	56.95%	1.072	63.59%	1.536	46.00%
Base + D(0.2) + DA	1.186	58.52%	1.073	63.39%	1.735	39.00%
Base + D(0.4) + DA	1.378	51.39%	1.207	59.29%	1.546	48.00%
Base + D(0.3) + DA + LR(0.01)	1.807	33.25%	1.745	35.07%	1.678	30.00%
Base + D(0.3) + DA + LR(0.001)	1.303	53.69%	1.13	61.92%	1.66	44.00%
Base + D(0.3) + DA + LR(0.0001)	1.613	43.54%	1.514	47.92%	1.467	47.00%
Base + D(0.3) + DA + LR(0.0001) + LF(15)	0.123	96.03%	0.157	95.02%	0.635	82.00%
Base + D(0.3) + DA + LR(0.00001) + LF(15)	0.177	94.33%	0.207	93.06%	0.9843	70.00%
Base + D(0.3) + DA + LR(0.0001) + LF(10)	0.092	97.32%	0.115	96.43%	0.623	87.00%

Table 2 Fine-tuned VGG-19 Network Performance

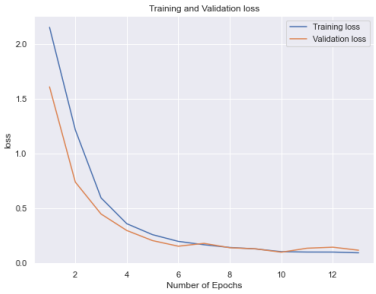

Figure 11 Loss

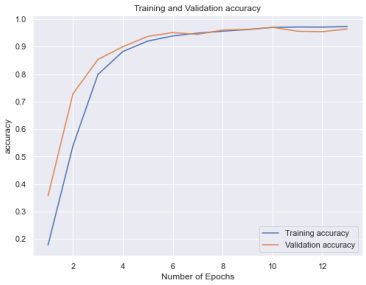

Figure 12 Accuracy

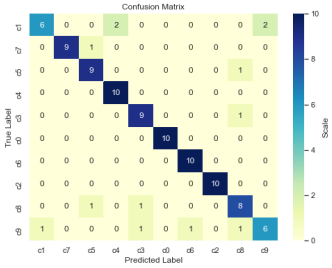

Figure 13 Confusion Matrix

5 CONCLUSION & LESSONS LEARNED

5.1 Conclusion

After evaluated the results for all trainings and performance of trained models, it is clearly that a model had been trained and is capable to detect distracted drivers from a dash camera image. Figure 14 shows the prediction result from a group of sample images. Therefore, we can conclude convolution neural network is an efficient approach for image classification problem. The results show fine-tuned VGG-19 network outperforms the convnet from scratch under the same maximum training epochs, batch size, and same input eventually. But this is not always true. Without re-train the convnet base, the scratch network always beats the fine-tuned and it takes less time to train, so scratch network seems to be a better choice from cost performance respective. Just like other neural networks, overfitting issue is a challenge for this project due to limited data, but the combination of image data augmentation, batch normalization and dropout minimized the impact of overfitting.

5.2 Lessons Learned

Throughout this project, there are mistakes were made which leads to un-expect result. For example, *horizontal_flip* was set to True while using *ImageDataGenerator*, which misleads the neural network since there is separation among left and right for distraction classes. I also learned that fine-tuning a pre-trained model needs to be look at case by case, good result is not guaranteed without re-training the convnet base, even though it may work very well for object detection tasks. In this project, partial layers from the convnet base must be set for trainable, and experiments are needed for determining number of layers to be re-trained. Besides, I also learned the Dropout rate does not show linear impact on the performance. Among 0.2, 0.3, and 0.4, 0.3 shows the best result. Lastly, I learned that there is no best learning rate for all networks. For example, with the same number of training epochs, 0.01 works very well for the scratch version convnet, while 0.001 is a better choice for fine-tuned VGG-19 convnet.

Figure 14 Prediction Result

6 FUTURE WORK

Based on the outcome of this project, the initial objectives have been met. However, there are still areas can be explored. With regarding to Image augmentation, this project only used basic image manipulations, which can be extended to use Deep learning approaches, such as adversarial training, neural style transfer, and GAN (Generative adversarial networks) Data augmentation [20]. Another area can be improved is the training platform, instead of relying on the CPU of personally MacBook Pro, it can use the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number ACI-1548562. Specifically, it used the Bridges-2 system, which is supported by NSF award number ACI-1445606, at the Pittsburgh Supercomputing Center (PSC). [5, 39] Further, trained model can be deployed to real environment, input images captured from cameras, for real-time detection, only as experimental study since the model was trained on a competition dataset.

ACKNOWLEDGMENTS

This project was inspired by driving education classes from Greg Driving School. I also would like to express sincere thanks to Dr. Liu, Xinlian for his great guidance given through the semester under the pandemic situation. In addition, I would like to give thanks to Omar Aboul-Enein for sharing his project with the class and providing tips that helps me to accomplish my project. Last not the least, I would like to express my deepest thanks my wife, my whole family and colleagues for their love, understanding, and support.

REFERENCES

[1] Centers Disease Control and Prevention (CDC) Transportation Safety. *Distracted Driving.* https://www.cdc.gov/transportationsafety/Distracted_Driving/index.html

[2] State Farm Distracted Driver Detection. https://www.kaggle.com/c/state-farm-distracted-driver-detection/data

[3] National Highway Traffic Safety Administration. (April 2020) Traffic Safety Facts Research Note: Distracted Driving 2018external icon. Department of Transportation, Washington, DC: NHTSA. Accessed 18 August 2020.

[4] "Deep Learning for Computer Vision." Deep Learning with Python, by Chollet François, Manning Publications Co., 2018.

[5] Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G.D., Roskies, R., Scott, J.R. and Wilkens-Diehr, N. 2014. XSEDE: Accelerating Scientific Discovery. Computing in Science & Engineering. 16(5):62-74. http://doi.ieeecomputersociety.org/10.1109/MCSE.2014.80.

[6] Omerustaoglu, F., Sakar, C. O., & Kar, G. (2020). Distracted driver detection by combining in-vehicle and image data using deep learning. Applied Soft Computing, 96, 106657.

[7] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res., 15(1):1929– 1958, Jan. 2014

[8] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, Journal of Machine Learning Research (JMLR) (2015) 448–456.

[9] Global status report on road safety. World Health Organization. https://www.who.int/violence_injury_prevention/road_safety_status/report/en

[10] Abouelnaga, Y., Eraqi, H. M., & Moustafa, M. N. (2017). Real-time Distracted Driver Posture Classification. December. http://arxiv.org/abs/1706.09498

[11] Streiffer, C., Raghavendra, R., Benson, T., & Srivatsa, M. (2017). DarNet: A deep learning solution for distracted driving detection. Middleware 2017 - Proceedings of the 2017 International Middleware Conference (Industrial Track), 22–28. https://doi.org/10.1145/3154448.3154452

[12] Rafael Berri and Alexandre Gonçalves Silva. A Pattern Recognition System for Detecting Use of Mobile Phones While Driving. (August), 2014. doi: 10.5220/0004684504110418.

[13] T Hoang Ngan Le, Yutong Zheng, Chenchen Zhu, Khoa Luu, and Marios Savvides. Multiple Scale Faster-RCNN Approach to Driver's Cell-Phone Usage and Hands on Steering Wheel Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 46–53, 2016. ISSN 21607516. doi: 10.1109/CVPRW.2016.13.

[14] C H Zhao, B L Zhang, J He, and J Lian. Recognition of driving postures by contourlet transform and random forests. IET Intelligent Transport Systems, 6(2):161–168, 2011a.

[15] Mofid, N., Bayrooti, J., & Ravi, S. (2020). Keep Your AI-es on the Road: Tackling Distracted Driver Detection with Convolutional Neural Networks and Targeted Data Augmentation. http://arxiv.org/abs/2006.10955

[16] Eraqi, H. M., Abouelnaga, Y., Saad, M. H., & Moustafa, M. N. (2019). Driver distraction identification with an ensemble of convolutional neural networks. Journal of Advanced Transportation, 2019. https://doi.org/10.1155/2019/4125865

[17] Tamas, V., & Maties, V. (2019). Real-Time Distracted Drivers Detection Using Deep Learning. American Journal of Artificial Intelligence, 3(1), 1. https://doi.org/10.11648/j.ajai.20190301.11

[18] Adwin Jahn. (2017) Keras Image Preprocessing: scaling image pixels for training. SpiderCloud Wireless, Inc. https://www.linkedin.com/pulse/keras-image-preprocessing-scaling-pixels-training-adwin-jahn/

[19] Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, 1–14.

[20] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. Journal of Big Data, 6(1). https://doi.org/10.1186/s40537-019-0197-0

[21] Xiao, J., Wang, J., Cao, S., & Li, B. (2020). Application of a Novel and Improved VGG-19 Network in the Detection of Workers Wearing Masks. Journal of Physics: Conference Series, 1518(1), 1–6. https://doi.org/10.1088/1742-6596/1518/1/012041

[22] Shallue, C. J., Lee, J., Antognini, J., Sohl-Dickstein, J., Frostig, R., & Dahl, G. E. (2019). Measuring the effects of data parallelism on neural network training. Journal of Machine Learning Research, 20, 1–49.

[23] Hoffer, E., Hubara, I., & Soudry, D. (2017). Train longer, generalize better: Closing the generalization gap in large batch training of neural networks. Advances in Neural Information Processing Systems, 2017-Decem, 1732–1742.

[24] Jetson Nano Developer Kit. https://developer.nvidia.com/embedded/jetson-nano-developer-kit

[25] Image data preprocessing. Keras. https://keras.io/api/preprocessing/image/

[26] ImageNet. http://image-net.org/

[27] Kumar, V., Choudhary, A., & Cho, E. (2020). Data Augmentation using Pre-trained Transformer Models. http://arxiv.org/abs/2003.02245

[28] Salman, S., & Liu, X. (2019). Overfitting Mechanism and Avoidance in Deep Neural Networks. http://arxiv.org/abs/1901.06566

[29] Arar, M., Danon, D., Cohen-Or, D., & Shamir, A. (2019). Image Resizing by Reconstruction from Deep Features. http://arxiv.org/abs/1904.08475

[30] Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. Insights into Imaging, 9(4), 611–629. https://doi.org/10.1007/s13244-018-0639-9

[31] Cogswell, M., Ahmed, F., Girshick, R., Zitnick, L., & Batra, D. (2016). Reducing overfitting in deep networks by decorrelating representations. 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings, 1–12.

[32] Lengerich, B., Xing, E. P., & Caruana, R. (2020). On Dropout, Overfitting, and Interaction Effects in Deep Neural Networks. 1–15. http://arxiv.org/abs/2007.00823

[33] May, P. (2019). Improved Image Augmentation for Convolutional Neural Networks by Copyout and CopyPairing. 1–8. http://arxiv.org/abs/1909.00390

[34] Stavrinos, D., Jones, J. L., Garner, A. A., Griffin, R., Franklin, C. A., Ball, D., Welburn, S. C., Ball, K. K., Sisiopiku, V. P., & Fine, P. R. (2013). Impact of distracted driving on safety and traffic flow. Accident Analysis and Prevention, 61, 63–70. https://doi.org/10.1016/j.aap.2013.02.003

[35] Kostrikov, I., Yarats, D., & Fergus, R. (2020). Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels. http://arxiv.org/abs/2004.13649

[36] Zhang, Q., Zhang, M., Chen, T., Sun, Z., Ma, Y., & Yu, B. (2019). Recent advances in convolutional neural network acceleration. Neurocomputing, 323, 37–51. https://doi.org/10.1016/j.neucom.2018.09.038

[37] Marshall, E. H. (1878). St. Paul and Roman law. Notes and Queries, s5-IX(229), 384–385. https://doi.org/10.1093/nq/s5-IX.229.384

[38] Perez, L., & Wang, J. (2017). The Effectiveness of Data Augmentation in Image Classification using Deep Learning. http://arxiv.org/abs/1712.04621

[39] Nystrom, N. A., Levine, M. J., Roskies, R. Z., and Scott, J. R. 2015. Bridges: A Uniquely Flexible HPC Resource for New Communities and Data Analytics. In Proceedings of the 2015 Annual Conference on Extreme Science and Engineering Discovery Environment (St. Louis, MO, July 26-30, 2015). XSEDE15. ACM, New York, NY, USA. http://dx.doi.org/10.1145/2792745.2792775.