Detection of Driver Drowsiness Using Multi-Task Learning

Yu-Hsuan, Tsai
Department of Computer Science and Information Engineering, National Taiwan University,
Taipei, Taiwan
Email: zfk.662624@gmail.com

Ming-Chun, Tseng
Department of Computer Science and Information Engineering, National Taiwan University,
Taipei, Taiwan
Email: tsngmj@gmail.com

Chiou-Shann, Fuh
Department of Computer Science and Information Engineering, National Taiwan University,
Taipei, Taiwan
Email: fuh@csie.ntu.edu.tw

Chin-Yung, Wang
Department of Electrical Engineering,
Rensselaer Polytechnic Institute,
New York
Email: wangc20@rpi.edu

Abstract

In recent studies of driver drowsiness detection, deep learning approaches have achieved success, but there are still some problems that the previous researches rarely considered. If we want to build a system that detects whether the mouth is yawning or not, we need to train a model that deals with this problem. On the other hand, we would need to build and train another system to detect whether the eyes are sleepy eyes or not. Previous studies have achieved success on the total score, which determines whether a driver is drowsy or not, but they would need another model to predict local information, such as the states of the eyes. In this paper, we think the relationship between the mouth, eyes, head, and drowsiness score is not negligible. For example, if a driver's mouth is in a state of yawning, then the eyes of the driver may be closed, therefore we can say the driver has sleepy-eyes. In light of these thoughts, we propose a new model to deal with this problem. We apply the concept of multi-task learning [1] and use Inception-ResNet-v2 [2] as the backbone for our CNN model. Finally, our approach achieved 92% accuracy on the head component, and 89% accuracy on the mouth component of the evaluation set of the public NTHU-DDD [3] dataset (National Tsuing-Hua University Driver Drowsiness Detection Dataset).

Keywords—Multi-Task Learning, Inception-ResNet-v2, Face Detector

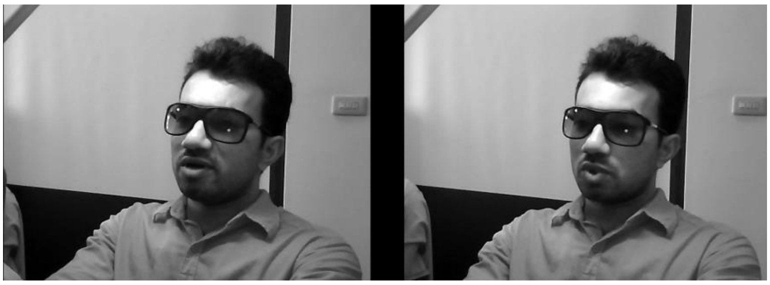
I. Introduction

The frequent occurrence of traffic accidents seriously threatens people's lives and property; therefore, the study of driver fatigue detection is of great significance. Driver drowsiness detection is a car safety technology which helps prevent accidents caused by the driver getting drowsy. Various studies have suggested that around 20% of all road accidents are fatigue-related, up to 50% on certain roads.

Various techniques already exist to measure driver fatigue. These techniques can generally be classified into three categories: vehicle-centric [4], driver-centric [5], and computer vision-based [6]. The first two methods to determine whether a driver is drowsiness are inconvenient. Computer vision-based method is possible to achieve the real-time detection and more convenient.

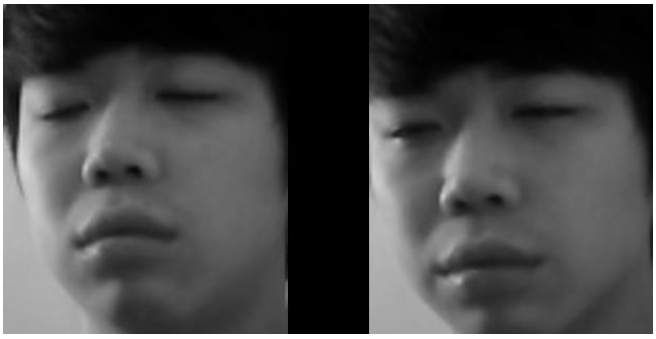
We use the public NTHU-DDD dataset as our training set and testing set. This dataset is widely used for researches in driver drowsiness detection. It consists of both male and female drivers, with various facial characteristics, different ethnicities, and five different scenarios. There are four kinds of annotations in each video and a single digit that is used to indicate the status of the frame: first, drowsiness – 0 means stillness, and 1 means drowsy. Second, head – 0 means stillness; 1 means nodding; and 2 means looking aside. Third, mouth – 0 means stillness; 1 means yawning; and 2 means talking and laughing. Last but not least, eyes – 0 means stillness, and 1 means sleepy-eyes.

The difficulty in this research is that it is hard to distinguish the difference between blinking eyes and sleepy eyes in Figure 1. It is also hard to distinguish the difference between a yawning mouth, a laughing mouth, or a talking mouth in Figure 2.



(a) (b)

Fig. 1. (a) Driver talking. (b) Driver possibly yawning.



(a) (b)

Fig. 2. (a) Driver blinking eyes. (b) Driver has sleepy eyes.

The previous studies have achieved success on the NTHU-DDD dataset, but there are still some problems that previous researches rarely considered. If we want to predict multiple states of the face parts at the same time, we would need to train multiple models and use these models to predict multiple outputs. We think the local parts on the face is relevant, and should not take them as the independent features.

We apply multi-task learning method in order to output multiple values by using the same model. The concept of multi-task learning is to output multiple values at the same time. By sharing representations between related tasks, we can generalize our model better on our original task, and reduce the risk of overfitting.

II. RELATED WORKS

Driver drowsiness is an important issue of road traffic safety. Many traditional approaches and deep learning approaches have achieved success in the driver drowsiness detection.

M. Omidyeganeh [7] provided a traditional method which detects the sequence of the mouth size to determine whether a driver is drowsy or not. T. H. Shih and C. T. Hsu [8] proposed a deep learning approach, using multi-stage spatial-temporal network to efficiently and accurately detect the driver drowsiness. They think spatial analysis is hard to determine whether a driver is drowsy, so they combine the CNN (Convolutional Neural Network) model with the Long Short-Term Memory (LSTM) model to analyze spatial and temporal features. They utilize CNN to extract the spatial feature of a frame, output a feature map, then take the feature map as the input of LSTM. LSTM analyzes the feature through time, thus can analyze the feature temporally and spatially. X. P. Huynh [9] proposed a 3-dimensional CNN model (3D-CNN). They utilize 3D convolutional neural network to extract features in spatial-temporal domain, and then utilize gradient boosting for drowsiness classification. Finally, utilize semi-supervised learning to enhance overall performance.

Multi-granularity Convolutional Neural Network (MCNN) was proposed by J. Lyu et al. [10]. MCNN is a novel network which utilizes a group of parallel CNN extractors on well-aligned facial patches of different granularities, and extracts facial representations effectively for large variation of head poses; furthermore, it can flexibly fuse both detailed appearance clues of the main parts and local to global spatial constraints, and achieve 90.05% accuracy on the drowsiness score of the NTHU-DDD. Convolutional Two-Stream Network was proposed by W. Liu et al. [11]. They utilize the gamma correction to enhance image contrast, then extract the static features from a partial facial image and dynamic features from a partial facial optical flow. Finally, they combine both static and dynamic features using a two-stream neural network for classification. They achieved 97.06% accuracy on the drowsiness score of the NTHU-DDD.

The studies mentioned above have achieved success on the detection of driver drowsiness, but they utilize one model to predict one output. We think the facial parts may be relevant. We can reuse some features, and make our model more general.

III. Our approach

Figure 3 illustrates how we detect the yawning and nodding state of a driver. First, we extract every 3 frames of the NTHU-DDD video, and store as images. We think it is unnecessary to analyze frame by frame because the features between adjacent frames are too close, and will produce almost the same data. Second, we detect the face in the frame by using the face detector, and thus we can crop the face part from image. Finally, we take the face part of the frame as the input of our model, then we can determine whether a driver is nodding or yawning at the same time.

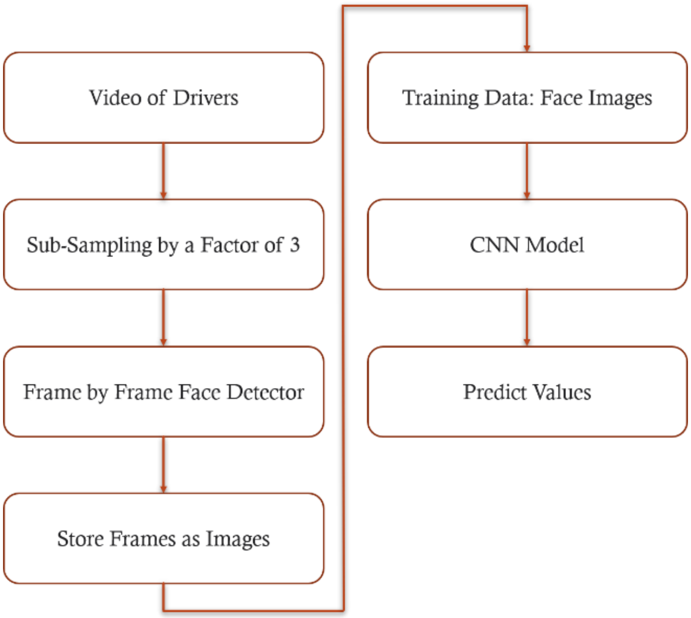

Video of Drivers
Sub-Sampling by a Factor of 3
Frame by Frame Face Detector
Store Frames as Images
Training Data: Face Images
CNN Model
Predict Values

Fig. 3. The flowchart of how we detect the yawning and nodding state of a driver.

A. *3x Sub-sampling the Video*

If the video is 30 fps (frame per second), then the interval time between frame to frame is 0.33s, and the feature of the adjacent frame is too close, so analyzing the information of every frame is unnecessary and only produces redundant computation.

B. *Face Detector*

After extracting the frame from the video, we need to crop out the face from the image, because the background object of the image might produce noise and make our model hard to focus on analyzing the feature on the face. We compare three different face detectors from different aspects. First, we compare the robustness of the three face detectors by counting the number of the miss-detection frames. Second, the computation time of an algorithm is very important, so we also compare the video fps of the three algorithms.

We first use OpenCV built-in Viola-Jones face detector [12] to detect the driver’s face from all images. The algorithm first uses different convolutional kernels to calculate many features called Haar Features. Then it boosts the algorithm using the integral image method. Finally, it classifies the face and the non-face using Adaboost [13] algorithm. The algorithm has a constraint which requires full view of the frontal upright faces. Thus in order to be detected, the entire face must point towards the camera and could not be tilted to either side.

Second, we apply the face detector of the dlib C++ library to detect the face in the image. This method uses a Maximum-Margin Object Detector (MMOD) [14] with CNN based features.

Finally, we apply the MTCNN [15] face detector algorithm to detect the face. MTCNN consists of three stages, P-Net, R-Net, and O-net, shown in Figures 4, 5, and 6. In the first stage, P-Net, produces candidate windows quickly through a shallow CNN. In the second stage, more complex CNNs refine the windows, rejecting a large number of non-faces and overlapping windows. In the third stage, more powerful CNNs were used to refine the result again, deciding whether the window should be rejected and find the five facial landmarks positions. In order to detect the face with different scales, the given image was scaled to different sizes to form an image pyramid. Moreover, the multi-task means that the output of P-Net will be the input of R-net, and the output of R-Net will be the input of O-net.

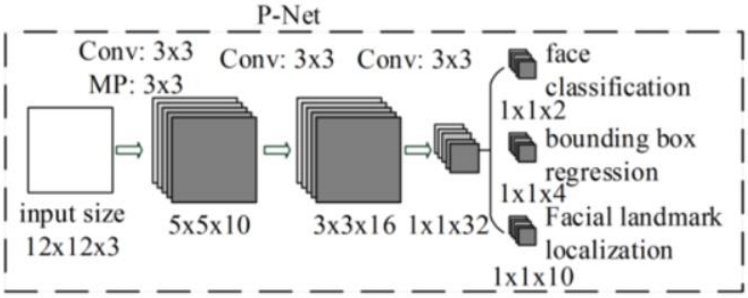


Fig. 4. The P-Net (Proposal Network) of the MTCNN.

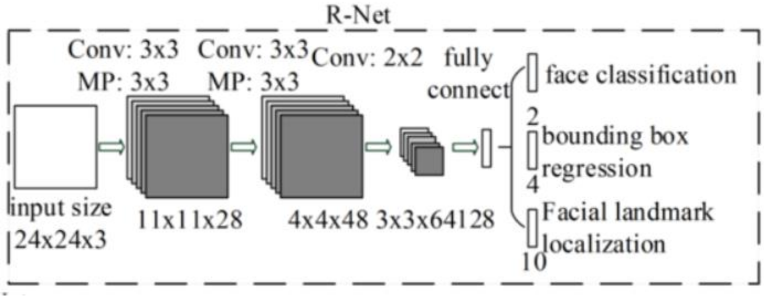


Fig. 5. The R-Net (Proposal Network) of the MTCNN.

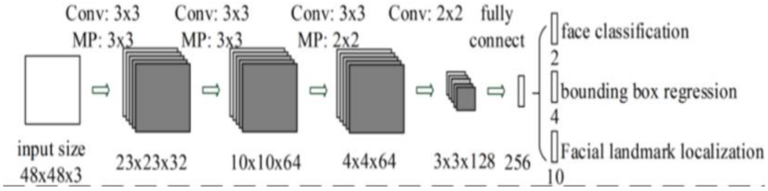


Fig. 6. The O-Net (Proposal Network) of the MTCNN.

Table I shows the number of miss-detection frames and the frames per second (fps) of the videos. The miss detection 1 and miss detection 2 is the missing face frame of the two videos in the NTHU-DDD dataset. The first video has 2,269 frames, there is no miss detection with MTCNN, 5 miss detection with dlib, and 424 miss detection with Viola-Jones face detector. The second video has 8,617 frames, MTCNN is the best face detector among the three again.

TABLE I. COMAPRSION BETWEEN THREE DIFFERENT FACE DETECTORS.

	Viola-Jones face detector	***Dlib***	***MTCNN***
Fps	100~150 fps	30~60 fps	30~50fps
Miss detection 1	424	5	0
Miss detection 2	987	999	292

We choose MTCNN as our face detector because of its robustness. We want to deal with the problem of head pose variation being too large, such as looking aside and lowering head. MTCNN can fulfill the requirements while maintain the efficiency. The face detector result is shown in Figure 7.

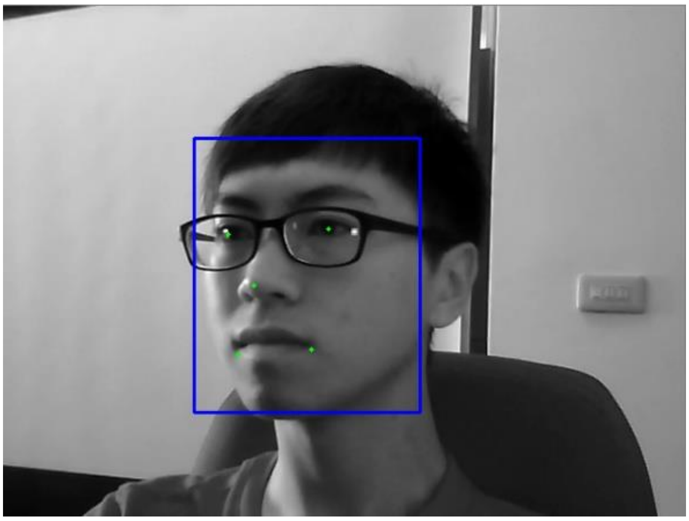


Fig. 7. The face detector result, we use the bounding box part as the input of CNN model.

C. *Multi-task Learning*

In the previous stage, we crop out the face part from a frame of a video. Now we can take it as the input of our CNN model to extract the spatial feature.

Multi-task learning is a subfield of machine learning in which multiple learning tasks are solved at the same time while exploiting commonalities and differences across tasks. This can result in improved learning efficiency and prediction accuracy for the task-specific models when compared with training the models separately.

The advantage of multi-task learning in our research is: First, we can use one model to predict two labels, and thus make the model size and parameter half than the normal research. Second, by sharing representations between related tasks, we can generalize our model better on our original task. Last but not least, parameter sharing greatly reduces the risk of overfitting. Learning just Task A bears the risk of overfitting Task A, while learning Tasks A and B jointly enables our model to obtain a better representation F through averaging the noise patterns.

The NTHU-DDD dataset has four annotations of each video and a single digit is used to indicate the status of the frame, eyes, mouth, head, and drowsiness. The normal research of driver drowsiness detection usually uses four different models to predict four different labels. When we want to predict the state of eyes, we need to train a model of eyes. When we want to predict the state of mouth, we need to train a model of mouth. We think the relationship of the mouth, eyes, head, and drowsiness score is important, so we propose a new method that can determine the state of different parts of a driver using the same model, that is, one model with multiple outputs.

Our model is shown in Figure 8. Our research only predicts two labels: mouth and head, because it is easy to analyze by only using spatial features, that is, the CNN model we used to extract spatial feature. We use Inception-ResNet-v2 as our CNN model backbone. The neural network combines Inception with Resnet, thus it has the advantage of both. Inception is an efficient deep neural network architecture. There is a simple but powerful way of creating better deep learning models, but can create complications. For example, it is more prone to overfitting, gradient vanishing, expensive computation, and so on. Inception is proposed to deal with these problems. ResNet utilizes skip connections, or short-cuts to jump over some layers, to avoid the problem of vanishing gradients, and reuse activations from a previous layer, which makes the network more powerful.

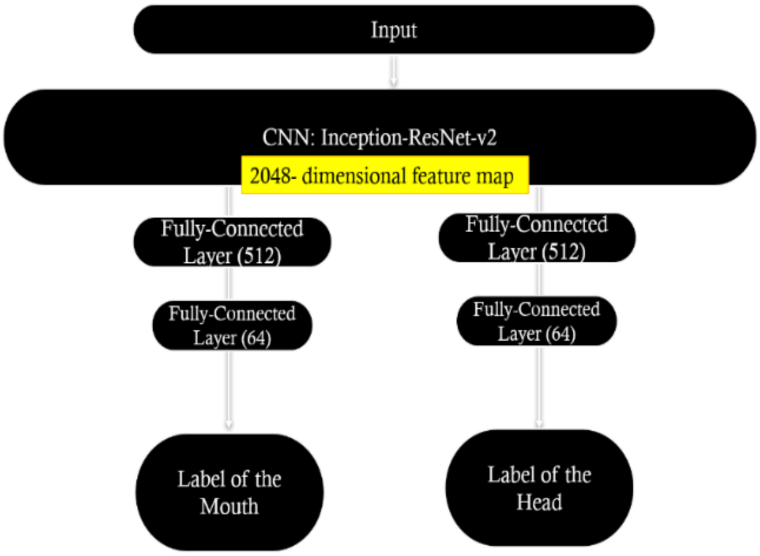

Input
CNN: Inception-ResNet-v2
2048- dimensional feature map
Fully-Connected Layer (512)
Fully-Connected Layer (512)
Fully-Connected Layer (64)
Fully-Connected Layer (64)
Label of the Mouth
Label of the Head

Fig. 8. Our detailed CNN model. The output of the Inception-ResNet-v2 model is a 2,048-dimensional feature map. Our research predicts two labels: mouth and head.

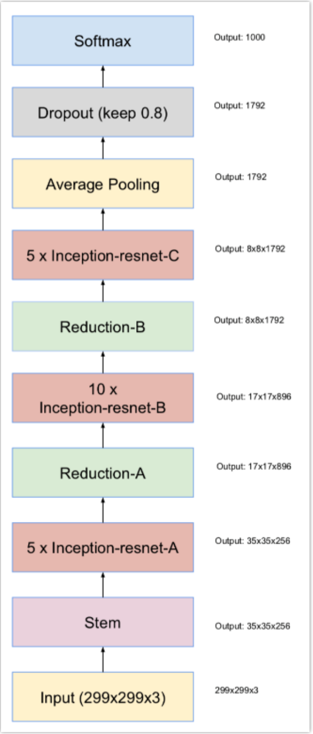


Fig. 9. The original version of Inception-ResNet-v2.

The time performance of our research is important. The entire architecture of Inception-ResNet-v2 is large and thus may require expensive computation. The entire network has 5x Inception-ResNet module C, 10x Inception-ResNet module B, and 5x Inception-ResNet module A in Figure 9. We decrease the number of each module, making our model with 2x Inception-ResNet module C, 2x Inception-ResNet module B, and 2x Inception-ResNet module A in Figure 10. Also, we keep the stem block and reduction block of Inception-ResNet-v2.

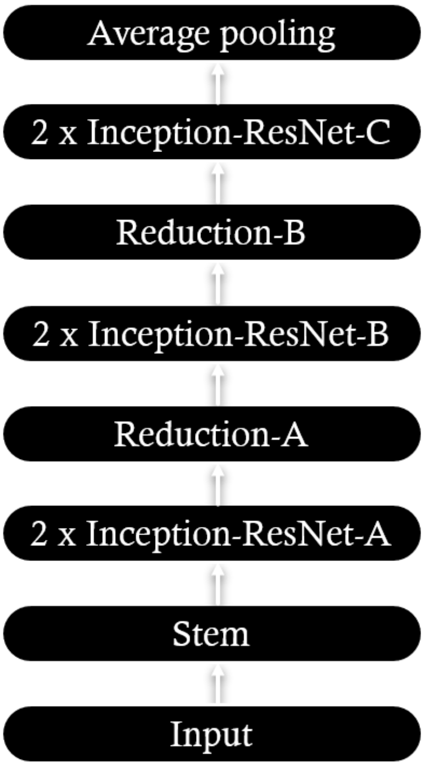


Fig. 10. We decrease the number of blocks and parameters in each module. Make our model has 2x Inception-ResNet module C, 2x Inception-ResNet module B, and 2x Inception-ResNet module A.

We apply the concept of multi-task learning on our Inception-ResNet-v2 model backbone. If we use one Inception-ResNet-v2 to predict one label, for example, mouth, then the model can only learn the feature of mouth, that is the feature map in the final layer of the model is all feature of mouth. We use one model to predict two labels: head and mouth to see whether a driver is nodding or looking aside, and whether a driver is yawning or talking. If the accuracy of both labels is high enough, we can say that the feature map in the final layer has the feature of mouth and head, because the state of the local parts is relevant. For example, if we want to distinguish between blinking eyes and sleepy eyes, we can use the feature of the mouth to support the model. If a driver is yawning, and the eyes of the driver are closed, we can say he has sleepy eyes because the state of his mouth is yawning.

IV. Experiments

A. *Implementation Details*

Preprocessing: In Figure 1, first, we extract every 3 frames of the NTHU-DDD video and crop the face part by using MTCNN face detector, then we resize the image to 66 x 85 pixels and take it as the input of the CNN model.

Face detector: We use MTCNN as the face detector. If the variation of head pose is too large, miss detection may occur. The solution to this problem is if the number of frames that face not be detected is less than 60, then we use the previous coordinate that face is detected because the face may not be far away from the coordinate. If the number of the frames that face not be detected is greater than 60, then we drop the frame because the time from the last frame that face be detected is too long.

CNN model: For the CNN model, that is, the modified Inception-ResNet-v2 model. We utilize the Adam optimizer to train the whole convolutional neural network with learning rate 0.0001. We also use random data augmentation to increase our data, also to prevent the overfitting. Our detailed CNN model is shown in Figure 3. The output of the Inception-ResNet-v2 model is a 2,048-dimensional feature map. We will then input them to two sets of separated fully-connected layers: one is to predict the state of the mouth and the other is to predict the state of the head. The set of the fully-connected layer consists of two hidden layers and an output layer. The meaning of the architecture is that if the final layer of the Inception-ResNet-v2 has the feature of the mouth and head, the following fully-connected layer supposed to be able to choose the feature it needs. For example, in the branch of predicting mouth, the fully-connected layer in the branch can choose the feature of the mouth and predict the state of the mouth.

Loss function: The loss functions of the head and mouth both utilize the categorical cross entropy, denoted as L_1 and L_2. The loss function we defined is weighted loss, that is:

$$Loss = L_1 + L_2$$

B. *Experiment result*

We use the testing set of NTHU-DDD dataset to evaluate the performance of our model. If we want to utilize the normal approach to predict two different labels in the same image, we need to train two different models to achieve the goal. The normal model we used to compare with is the same as the model we proposed. Both normal models (head and mouth) use the Inception-ResNet-v2 as the backbone while connecting to fully-connected layer to classify the status of the head or the mouth.

The experiment result, shown in Table II, is the accuracy of the two different labels on evaluation set of NTHU-DDD using the multi-task learning and the single-task learning. There are 3 statuses to represent the state of a driver's head and also 3 statuses to represent the state of a driver's mouth. We can achieve almost the same, even better performance by using one model to predict multi-outputs compared with the single-task learning, that is, one model predicts one output value. That means if a user wants to know if the mouth is yawning and if the head is nodding, the inference time of our model is shorter, and meanwhile can guarantee the high accuracy.

TABLE II. THE ACCURACY OF THE TWO DIFFERENT LABELS ON EVALUATION SET OF NTHU-DDD BY MULTI-TASK LEARNING AND SINGLE-TASK LEARNING. MULTI-TASK LEARNING USES ONE MODEL TO PREDICT HEAD AND MOUTH LABELS. SINGLE-TASK LEARNING REQUIRES TWO MODELS TO PREDICT HEAD AND MOUTH LABELS, THUS LESS EFFICIENT.

	Multi-task learning	***Single-task learning***
head	92%	93%
mouth	89%	88%

V. CONCLUSION

The disadvantage of the NTHU-DDD is: First, the instant of the drowsiness is subjective, it is hard to indicate the frame that the state of a driver changes from awake to drowsiness precisely. Second, the dataset is simulated in the laboratory, the dataset lacks the situation that brightness changes abruptly. This may occur in real-life scenario, for instance, when a driver drives through the tunnel, the brightness changes quickly, and thus makes the feature different from the scenario in the laboratory. Furthermore, the shake or bump of the car is also ignored. This may be the future work we can do to further achieve a robust system.

Our novel approach to detect the drowsiness state of a driver uses the same model to predict different labels in the same image. In other word, better efficiency with detecting the drowsiness state of a driver. Because the safety of a driver is important, and it may need to achieve real-time speed, how to get the better efficiency is an important issue. Also, using multi-task learning to train a model guarantees that the feature map of the final layer is actually the feature we want. For example, if we train a binary-classification model by using only drowsiness label, and the drowsiness score is highly relevant to the state of mouth, the model can easily achieve high accuracy by extracting only the feature of the mouth, and thus make the model only learn the feature of the mouth. But if we use the multi-task learning method to train a model, a model will output the status of the head, the eyes, and the head, then in order to achieve the best performance for the four labels at the same time, the model will learn the feature of the eyes, mouth, and the head. Thus make the model more general. Furthermore, we define a weighted loss function to combine the loss of two different labels.

Finally, we achieve almost the same, even the better performance by using one model to predict multiple outputs.

References

[1] Caruana, R., "Multi-Task Learning," *Machine Learning*, Vol. 28, pp. 41–75, 1997.

[2] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, Inception-Resnet and the Impact of Residual Connections on Learning," *Proceedings of International Conference on Learning Representations Workshop*, San Juan, Puerto Rico, pp. 1-12, 2016.

[3] C. H. Weng, Y. H. Lai, and S. H. Lai, "Driver Drowsiness Detection via a Hierarchical Temporal Deep Belief Network," *Proceedings of Asian Conference on Computer Vision Workshop on Driver Drowsiness Detection from Video*, Taipei, Taiwan, pp. 117-133, 2016.

[4] A. Colic, O. Marques, and B. Furht, *Driver Drowsiness Detection - Systems and Solutions*, ser. Springer Briefs in Computer Science. Springer, 2014.

[5] J. Wang and Y. Gong, "Recognition of Multiple Drivers' Emotional State," *Proceedings of IEEE International Conference Pattern Recognition*, Tampa, Florida, USA, pp. 1–4, 2008.

[6] T. Nakamura, A. Maejima, and S. Morishima, "Detection of Driver's Drowsy Facial Expression," *Proceedings of Asian Conference on Pattern Recognition*, Naha, Japan, pp. 749–753, 2013.

[7] M. Omidyeganeh, S. Shirmohammadi, S. Abtahi, A. Khurshid, M. Farhan, J. Scharcanski, B. Hariri, D. Laroche, and L. Martel, "Yawning Detection Using Embedded Smart Cameras," *IEEE Transactions on Instrumentation and Measurement*, Vol. 65, No. 3, pp. 1-13, 2016.

[8] T. H. Shih and C. T. Hsu, "MSTN: Multistage Spatial-Temporal Network for Driver Drowsiness Detection," *Proceedings of Asian Conference on Computer Vision*, Taipei, Taiwan, pp. 146-153, 2016.

[9] X. P. Huynh, S. M. Park, and Y. G. Kim, "Detection of Driver Drowsiness Using 3D Deep Neural Network and Semisupervised Gradient Boosting Machine," *Proceedings of Asian Conference on Computer Vision Workshop*, Taipei, Taiwan, pp. 134-145, 2016.

[10] J. Lyu, Z. Yuan, and D. Chen, "Long-term Multi-granularity Deep Framework for Driver Drowsiness Detection," https://arxiv.org/abs/1801.02325, 2018.

[11] W. Liu, J. Qian, Z. Yao, X. Jiao, and J. Pan, "Convolutional Two-Stream Network Using Multi-Facial Feature Fusion for Driver Fatigue Detection," *Future Internet*, Vol. 11, No. 115, pp. 1-13, 2019.

[12] P. Viola and M. Jones, "Robust Real-Time Object Detection," *International Journal of Computer Vision*, Vol. 57, pp. 137–154, 2004.

[13] Y. Freund and R. E. Schapire, "A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting," *Journal of Computer and System Sciences*, Vol. 55, pp. 119-139, 1997.

[14] D. E. King, "Max-margin Object Detection," https://arxiv.org/abs/1502.00046, 2015.

[15] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint Face Detection and Alignment Using Multi-Task Cascaded Convolutional Networks," *IEEE Signal Processing Letters*, Vol. 23, No. 10, pp. 1499-1503, 016.