Toward Extremely Lightweight Distracted Driver Recognition With Distillation-Based Neural Architecture Search and Knowledge Transfer

Dichao Liu, Toshihiko Yamasaki, *Member, IEEE*, Yu Wang, *Member, IEEE*, Kenji Mase, and Jien Kato, *Senior Member, IEEE*

***Abstract*—The number of traffic accidents has been continuously increasing in recent years worldwide. Many accidents are caused by distracted drivers, who take their attention away from driving. Motivated by the success of Convolutional Neural Networks (CNNs) in computer vision, many researchers developed CNN-based algorithms to recognize distracted driving from a dashcam and warn the driver against unsafe behaviors. However, current models have too many parameters, which is unfeasible for vehicle-mounted computing. This work proposes a novel knowledge-distillation-based framework to solve this problem. The proposed framework first constructs a high-performance teacher network by progressively strengthening the robustness to illumination changes from shallow to deep layers of a CNN. Then, the teacher network is used to guide the architecture searching process of a student network through knowledge distillation. After that, we use the teacher network again to transfer knowledge to the student network by knowledge distillation. Experimental results on the Statefarm Distracted Driver Detection Dataset and AUC Distracted Driver Dataset show that the proposed approach is highly effective for recognizing distracted driving behaviors from photos: (i) the teacher network's accuracy surpasses the previous best accuracy; (ii) the student network achieves very high accuracy with only 0.42M parameters (around 55% of the previous most lightweight model). Furthermore, the student network architecture can be extended to a spatial-temporal 3D CNN for recognizing distracted driving from video clips. The 3D student network largely surpasses the previous best accuracy with only 2.03M parameters on the Drive&Act Dataset. The source code is available at https://github.com/Dichao-Liu/Lightweight_Distracted_Driver_Recognition_with_Distillation-Based_NAS_and_Knowledge_Transfer**

Manuscript received 3 April 2022; revised 24 August 2022; accepted 27 September 2022. Date of publication 7 November 2022; date of current version 26 January 2023. This work was supported in part by the Ph.D. Professional Toryumon Program. The Associate Editor for this article was Y. Kamarianakis. *(Corresponding author: Dichao Liu.)*

Dichao Liu was with the Graduate School of Informatics, Nagoya University, Nagoya 464-8601, Japan. He is now with Navier, Inc., Tokyo 102-0084, Japan (e-mail: dichao_liu@outlook.jp).

Toshihiko Yamasaki is with the Graduate School of Information Science and Technology, The University of Tokyo, Tokyo 113-8656, Japan.

Yu Wang was with the College of Information Science and Engineering, Ritsumeikan University, Kusatsu-shi, Shiga 525-8577, Japan. He is now with the Center for Information and Communication Technology, Hitotsubashi University, Tokyo 186-8601, Japan.

Kenji Mase is with the Graduate School of Informatics, Nagoya University, Nagoya 464-8601, Japan.

Jien Kato is with the College of Information Science and Engineering, Ritsumeikan University, Kusatsu-shi, Shiga 525-8577, Japan.

Digital Object Identifier 10.1109/TITS.2022.3217342

***Index Terms*—Distracted driving, decreasing filter size, advanced driver assistance, intelligent vehicles, ConvNets, action recognition.**

I. Introduction

As DEFINED by the National Highway Traffic Safety Administration in the United States (NHTSA), distracted driving is "any activity that diverts attention from driving" [1], [2], such as drinking, talking to passengers, etc. Nowadays, distracted driving has become a huge threat to modern society. For example, as reported by the NHTSA, in the United States, traffic accidents caused by distracted driving led to 3,142 or 8.7 percent of all accidents in 2019 [3].

Recently, Advanced Driver Assistance Systems (ADAS) are being developed to provide technologies that alert the driver to potential problems for preventing accidents. As one of the basic and most important technologies of ADAS, distracted driver recognition (DDR) has attracted much interest from the academic society [4], [5], [6], [7]. Many approaches have been developed to use the images taken by a dashcam to recognize whether the driver is driving safely or behaving some categories of distracted driving actions [8], [9], [10], [11], [12], [13]. With the effort of the researchers, the recognition accuracy of the DDR task has been increasing, especially when convolutional neural networks (CNNs) are employed in this field [8], [10], [14], following the success of CNNs in many other fields. However, the accuracy improvement is generally brought by increased CNN parameter size. The huge parameter size becomes a big problem for real-world applications because of the limitation of vehicle-mounted computing equipment. The purpose of this paper is to design a lightweight and fast network for DDR with high DDR accuracy, which will be very useful for intelligent transportation system (ITS) applications. In the remainder of this section, we start with a review of the existing DDR methods and then briefly present a general overview of our approach.

A. Existing Distracted Driver Recognition Approaches

Recently, with the success of CNNs in the computer vision field, it has become common to use deep learning models to solve distracted driver recognition (DDR) tasks [8], [15], [16].

For example, Yan et al. [16] embedded local neighborhood operations and trainable feature selectors within a deep CNN, and by doing so, meaningful features could be selected automatically to recognize distracted drivers.

However, the introduction of CNNs causes the problem of huge parameter size. There are some recent lightweight networks designed for general-purpose computer vision, such as MobileNet [17], MobileNetV2 [18] and SqueezeNet [19]. However, these lightweight networks are not specifically designed for DDR, and therefore is still room for improvement regarding DDR accuracy and the number of parameters.

There are now also some lightweight networks designed specifically for DDR by hand. For example, Baheti et al. [20] propose the MobileVGG, which reduces the number of parameters by replacing the traditional convolution in the classical VGG structure with depth-wise convolution and point-wise convolution. D-HCNN [21] is another example, which uses an architecture containing four convolution blocks with the filters of rather large spatial sizes and achieves high performance with small number of filters. However, these networks were designed entirely by hand based on experience with networks used for general-purpose computer vision tasks, so the potential of the network structure could not be reached to the maximum extent possible. Moreover, D-HCNN requires histogram of oriented gradients (HOG) [22] in addition to RGB image as the input. HOG counts occurrences of gradient orientation in localized portions of an image and describes the appearance and shape of the local objects. The computation of HOG requires extra processing effort and is not favorable for real-world applications.

In this work, we search for an optimal architecture for the DDR task, which has less parameter size and higher accuracy than the above studies. Our approach is designed by NAS rather than totally by hand and only requires RGB images as inputs.

B. A Brief Overview of the Proposed Approach

To solve this problem, we propose a distillation-based neural architecture search and knowledge transfer framework. Overall, the proposed framework is based on knowledge distillation [23], which refers to the process of transferring knowledge from a large model (teacher network) to a smaller one (student network). The proposed framework includes three steps: (i) constructing a strong teacher network; (ii) searching and define the architecture of a student network under the supervision of the teacher network; (iii) transferring the knowledge from the teacher network to the student network.

1) Teacher Network: The teacher network is built based on progressive learning (PL). PL is a training strategy that starts the training from shallow layers and then progressively deepens the model by adding new layers to the model [8], [15], [16]. In some studies, PL is also regarded as partitioning a network into several segments and progressively training the segments from shallow to deep [25], [26]. Progressive learning (PL) was originally proposed for generative adversarial networks [27]. It started with low-resolution images, and then progressively increased the resolution by adding layers to the networks. For example, Wang et al. [28] proposed to progressively cascade residual blocks to increase the stability of processing extremely low-resolution images with very deep CNNs. Shaham et al. [29] proposed to reconstruct high-resolution images by a progressive multi-scale approach that progressively up-sample the output from the previous level. Recently, PL has been also applied in fine-grained image classification tasks. For example, Du et al. [25] and Zhao et al. [26] used PL to fuse information from previous levels of granularity and aggregate the complementary information across different granularities.

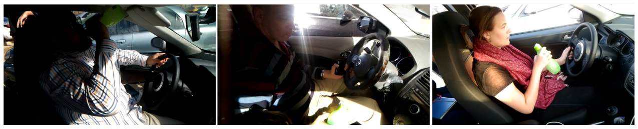


Fig. 1. Examples of images taken by a camera monitoring the driver's behavior under different illumination conditions. The ground-truth label of the images is "Drink". The images are from the AUC Distracted Driver Dataset [24].

In this work, we introduce PL into DDR to solve the problem caused by various illumination conditions, such as sunlight and shadow. As shown in Figure 1, in the real world, the dashcam commonly records the driver's behavior in different illuminations, while the color itself is susceptible to the influence of illumination. RGB information changes considerably under different illuminations, which causes strong intra-class variance in the DDR task. Such intra-class variance affects CNNs from shallow to deep layers. The shallow layers of CNN tend to learn basic patterns, such as different orientations, parallel lines, curves, circles, etc., while the deep layers tend to encode the patterns learned by shallow layers to capture more semantically meaningful information, such as hands, body, etc [30]. Following the experience learned with bright illumination on what basic patterns are discriminative, the shallow layers of a CNN might fail to find enough discriminative basic patterns in the shadows.

In this work, we progressively train the teacher network for several stages. During the stages, the training starts from shallow layers and progressively goes deeper with random brightness augmentation [31] to increase the robustness to the illumination of the layers from shallow to deep. Thereafter, we use the original image to train the aggregation of the models of all stages, considering that the random brightness augmentation might lose some visual information.

2) Student Network: The student network is a compact network that should be able to achieve high recognition performance. This leads to a research question: how to define the architecture of the student network to make it compact, lightweight, yet powerful for DDR, by utilizing the knowledge of the teacher network as supervision?

To answer this question, we turn our eyes to neural architecture search (NAS). NAS refers to the process of automating architecture engineering to learn a network topology that can achieve best performance on a certain task [32], [33], [34].

The major components of NAS includs searching space, searching algorithm, and evaluation strategy [32]. With the prior knowledge about typical properties of architectures, NAS approaches commonly define the searching space as a large set of operations (e.g., convolution, fully-connected, and pooling). Each possible architecture in the searching space is evaluated by a certain evaluation strategy [32], [33] and the searching process is controlled by certain searching algorithms, such as reinforcement learning [33], [35], [36], evolutionary search [37], differentiable search [38], or other learning algorithms [34], [39], [40], [41]. NAS commonly defines a searching space at first and then uses a certain policy to generate a sequence of actions in the searching space to specify the architecture.

In this work, we propose a new searching approach for DDR based on the characteristics of the images in the DDR task. We introduce how we define the searching space and the searching strategy as described below.

3) Searching Space: The images in the DDR task have less diversity and much stronger inter-class similarity than those in many other image recognition tasks. For example, in the fine-grained image recognition task of CUB Birds [42], the images contain the birds of different species, the background of different habitats, etc. However, in the DDR task, almost all the images can be roughly described as “a human is driving.” Thousands of images showing different driving behaviors might be performed by the same person, and the backgrounds of all the images are actually the interior of the same car.

Due to the above reason, a large proportion of the visual information does not provide discriminative clues in the DDR task. For example, in CUB Birds, the color of wings, the shape of heads, etc. all provide useful information. Sometimes, even the background provides useful information as a bird image with the sea as the background highly likely shows a certain sea bird. In contrast, in the DDR task, the color of the driver’s clothes, the shape of the driver’s glasses or hat, almost all the background, etc. are useless information.

Consequently, the models for the DDR task do not need a huge number of object detectors. The key is to explore some discriminative objects, which are quite universal among different driving behaviors, such as hands, body pose, steering wheel, etc. In CNNs, depth influences the flexibility, and each channel of the filters acts as an object detector [30]. Thus, the architecture for DDR does not require a very deep structure and a huge number of channels. The above claim is backed up by some earlier observations that the architecture of a decreased number of layers and channels can achieve good results in DDR [20], [21].

On the other hand, the architecture for DDR must be able to effectively find and capture useful clues from the limited discriminative objects, which is very difficult because: (i) the inter-similarity is strong; (ii) the key objects vary largely in size (e.g., hands and body). In this work, we introduce pyramidal convolution (PyConv) [43] into the DDR task. In a standard convolution layer, all the filters have the same spatial size. In contrast, a PyConv layer uses convolution filters of different spatial sizes, and the filters are possible to divide into several groups. Thus, PyConv has very flexible receptive fields, which is beneficial to capture key objects of different sizes. Also, due to its flexibility, PyConv provides a large pool of potential network architectures. In this work, the main searching space is defined as the candidate combinations of filters’ spatial sizes and the number of groups. Moreover, the pooling method applied in the model also influences the performance of capturing key objects [44]. We also search whether to use max pooling or average pooling in the layers.

4) Searching Strategy: Most of the NAS methods train the possible candidate networks one by one, and evaluate the performance of the trained candidate networks on a validation set [32], [45]. The evaluation results are used as metrics to update the architecture searching process. However, the process of candidate evaluation could be very expensive in terms of time, memory, computation, etc. In this work, since we have already constructed a powerful teacher network, we directly use the teacher network to guide the searching. Specifically, we first build a super student network that aggregates all the candidates with a weighted sum, whose weights are regarded as the possibility of choosing each candidate. Then the super student network is trained to learn from the teacher network by knowledge distillation. After the training, the candidates with the maximum weight are chosen to build the architecture of the student network.

After defining the architecture of the student network, the teacher network is utilized again to transfer knowledge to the student network.

Our contributions are summarized as follows:

- We propose a novel framework for solving the DDR task with high accuracy and a small number of parameters. The research question is solved by the proposed searching strategy.
- We mainly carried out the experiments of training the teacher network, defining the student network, and evaluating the performance of the teacher and student networks on two image-based DDR datasets, namely the AUC Distracted Driver Dataset (AUCD2) [24] and State-farm Distracted Driver Detection Dataset (SFD3) [46]. The experimental results show that the teacher network achieves 96.35% on the AUCD2 and 99.86%–99.91% in different splitting settings on the SFD3 with 44.62M parameters, which outperforms the previous state-of-the-art approaches on both datasets. Note that the previous best approach on AUCD2 requires 140M parameters.
- The student network achieves 95.64% on the AUCD2 and 99.86%–99.91% in different splitting settings on the SFD3 with only 0.42M parameters.
- The student network architecture can be extended into a spatial-temporal 3D convolutional neural network by replacing the 2D layers with spatial-temporal 3D layers [47], [48], [49], [50]. We carried out comprehensive experiments in all the tasks of the Drive&Act Dataset (DAD) [51], which is a video-based DDR dataset. The 3D student network is 0.89%–29.00% higher than the previous best accuracy in the validation set and 2.05%–30.88% higher than the previous best accuracy in the test set. The 3D student network requires only 2.03M parameters.

II. Details of the Proposed Approach

A. Teacher Network Construction

In this subsection, we introduce the details of the teacher network. Let E be the backbone feature extractor, which can be based on any state-of-the-art models, such as SKResNeXt50 [52], etc. The layers of E are divided into N segments $\{m_1, m_2, \ldots, m_n, \ldots, m_N\}$. Assume $\{s_1, s_2, \ldots, s_n, \ldots, s_N\}$ be N consecutive stages from shallow to deep. At each stage of $\{s_1, s_2, \ldots, s_n, \ldots, s_N\}$, the training always starts from the first layer of E. From s_1 to s_N, the training gradually goes deeper and covers more layers of E. That is, the segments under training at stage s_n are $m_1 + m_2 + \ldots + m_n$. Let $\{x_1, x_2, \ldots, x_n, \ldots, x_N\}$ denote the the output feature maps at $\{s_1, s_2, \ldots, s_n, \ldots, s_N\}$. Let $x_n \in \mathbb{R}^{H_n \times W_n \times C_n}$ denote the output feature map at the stage s_n, and H_n, W_n, and C_n respectively denotes the height, width, and the number of channels of x_n. We use a set of operations $\{\phi_1(.), \phi_2(.), \ldots, \phi_n(.), \ldots, \phi_N(.)\}$ to respectively process $\{x_1, x_2, \ldots, x_n, \ldots, x_N\}$ into 1D vectorial descriptors $\{v_1, v_2, \ldots, v_n, \ldots, v_N\}$, where $v_n \in \mathbb{R}^L$. The $\phi_n(.)$ corresponding to x_n is defined as:

$$v_n = \phi_n(x_n) = f^{\text{max\_pool}}_{H \times W}(x''_n), \tag{1}$$

$$x''_n = f^{\text{ReLU}}(f^{\text{bn}}(f^{\text{conv}}_{3\times3\times\frac{L}{2}\times L}(x'_n))), \tag{2}$$

$$x'_n = f^{\text{ReLU}}(f^{\text{bn}}(f^{\text{conv}}_{1\times1\times C\times\frac{L}{2}}(x_n))), \tag{3}$$

where $f^{\text{max\_pool}}_{H \times W}(.)$ denotes a max-pooling operation whose window size is $H \times W$. $f^{\text{conv}}(.)$ illustrates the 2D convolution operation by kernel size. For example, $f^{\text{conv}}_{1\times1\times C\times\frac{L}{2}}(.)$ denotes a 2D convolution operation whose kernel size is $1 \times 1 \times C \times \frac{L}{2}$ (1×1 is the spatial size, C is the number of input channels, and $\frac{L}{2}$ is the number of output channels). $f^{\text{bn}}(.)$ denotes the batch normalization operation [53], and $f^{\text{ReLU}}(.)$ denotes the ReLU operation.

Thereafter, we use a set of operations $\{\psi_1(.), \psi_2(.), \ldots, \psi_n(.), \ldots, \psi_N(.)\}$ to respectively process $\{v_1, v_2, \ldots, v_n, \ldots, v_N\}$ to predict the probability distribution $\{p_1, p_2, \ldots, p_n, \ldots, p_N\}$ over the classes at each stage:

$$\begin{aligned} p_n &= \psi_n(v_n) \\ &= f^{\text{fc}}_{\frac{L}{2}\times K}(f^{\text{ReLU}}(f^{\text{bn}}(f^{\text{fc}}_{L\times\frac{L}{2}}(f^{\text{bn}}(v_n))))), \end{aligned} \tag{4}$$

where $p_n \in \mathbb{R}^K$, and K denotes the number of the classes of driving behaviors. $f^{\text{fc}}_{\frac{L}{2}\times K}(.)$ denotes a fully connected layer whose input size is $\frac{L}{2}$ and the output size is K. $f^{\text{fc}}_{L\times\frac{L}{2}}(.)$ denotes a fully connected layer whose input size is L and the output size is $\frac{L}{2}$.

After the last stage s_n, we add an additional stage by concatenating $v_1, v_2, \ldots, v_N$ and generating the concatenated vector into the probability distribution over the classes as:

$$\begin{aligned} p_{N+1} &= \psi_{N+1}(v_{N+1}) \\ &= f^{\text{fc}}_{\frac{L}{2}\times K}(f^{\text{ReLU}}(f^{\text{bn}}(f^{\text{fc}}_{NL\times\frac{L}{2}}(f^{\text{bn}}(v_{N+1}))))), \end{aligned} \tag{5}$$

$$v_{N+1} = f^{\text{concat}}(v_1, v_2, \ldots, v_n, \ldots, v_N), \tag{6}$$

where $f^{\text{concat}}(.)$ denotes the concatenation operation. Now, we have $N + 1$ prediction probability distributions $\{p_1, p_2, \ldots, p_n, \ldots, p_N, p_{N+1}\}$. The teacher network is trained by using a cross entropy loss $\mathcal{L}_{\text{cls}}(.)$ to minimize the distance between ground truth label p_{truth} and each prediction probability distribution of $\{p_1, p_2, \ldots, p_n, \ldots, p_N, p_{N+1}\}$:

$$\mathcal{L}_{\text{cls}}(p_n, p_{\text{truth}}) = -\sum_{k=1}^{K} p^{(k)}_{\text{truth}} \log(p^{(k)}_n), \tag{7}$$

where $p^{(k)}_n$ denotes the probability that the input belongs to the category k at the stage s_n. $p^{(k)}_{\text{truth}}$ equals to 1 if it is true that the input belongs to the category k, and equals to 0 on the contrary.

Algorithm 1 Building the Teacher Network Based on Progressive Learning

Require: Given a dataset $\mathcal{D} = \{(\text{input}^i,\ p^i_{\text{truth}})\}^I_{i=1}$ (I is the total number of images in $\mathcal{D}$), and N stages $\{s_1, s_2, \ldots, s_n, \ldots, s_N\}$ of the backbone feature extractor E.

```
for epoch ∈ [1, num_of_epoch] do
  for (input, $p_{\text{truth}}$) in $\mathcal{D}$ do
    for $n \in [1, N]$ do
      $\text{input}_\text{n}$ = Brightness_augmentor(input)
      $x_n = s_n(\text{input}_n)$
      $v_n = \phi_n(x_n)$
      $p_n = \psi_n(v_n)$
      $\mathcal{L}_n = \mathcal{L}_{\text{cls}}(p_n, p_{\text{truth}})$
      BACKPROP($\mathcal{L}_n$)
    end for
    for $n \in [1, N]$ do
      $\text{input}_n = \text{input}$
      $x_n = s_n(\text{input}_n)$
      $v_n = \phi_n(x_n)$
    end for
    $v_{N+1} = f^{\text{concat}}(v_1, v_2, \ldots, v_N)$
    $p_{N+1} = \psi_{N+1}(v_{N+1})$
    $\mathcal{L}_{N+1} = \mathcal{L}_{\text{cls}}(p_{N+1}, p_{\text{truth}})$
    BACKPROP($\mathcal{L}_{N+1}$)
  end for
end for
```

The overall algorithm of building the teacher network is given in Algorithm 1. For the stages $s_1 \sim s_n$, the input images are augmented with Imgaug [31].

B. Distillation-Based Neural Architecture Search for the Student Network

The computation overhead, including speed and parameter size, acts as an extremely crucial role for DDR. According to the experiences of previous studies [12], [21], it is much more favorable to use large convolution filters rather than deep layers because the former is able to compute in parallel to achieve a fast processing speed that satisfies the requirements of the real-world application. Thus, in this work, we design the student network to have four convolutional blocks, which are

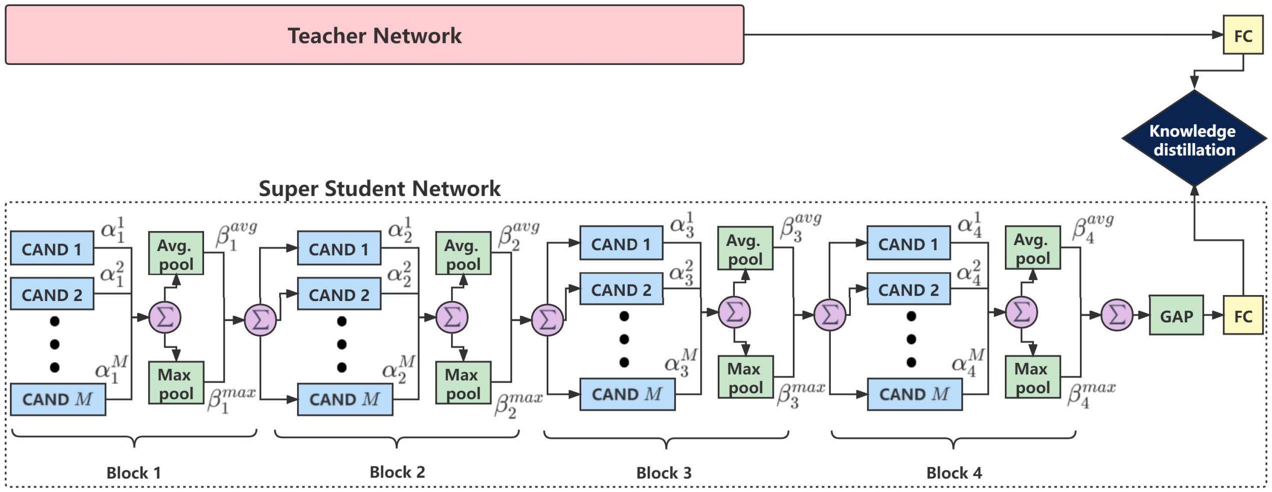


Fig. 2. Illustration of the searching process. "CAND" is the abbreviation for "candidate". In the super student network, there are several candidates for the convolutional architectures of each block. Besides, there are two candidates of the pooling method, namely average pooling and max pooling, in each block. The candidates of convolutional architecture and pooling methods for each block are aggregated by the weighted sum. α and β are the learnable weights. "GAP" means global average pooling and "FC" means the fully-connected operation. The super student network is trained to learn from the teacher network by knowledge distillation. After the training, only the candidates with the maximum weight are kept and forms the student network.

followed by a global average pooling layer (GAP) and a fully connected (FC) layer for predicting the probability distribution over the classes.

For each block, we use pyramidal convolution (PyConv) [43] rather than standard convolution. PyConv contains a pyramid of kernels, where each level involves different types of filters with varying sizes. Using PyConv for DDR has two benefits. First, PyConv can capture different levels of details in the scene. A filter of a smaller kernel size has smaller receptive fields and thus can capture more local information and more detailed clues. A filter of a bigger kernel size has bigger receptive fields and thus can "see" more information at once and capture relatively more global information, such as the dependencies among some local patterns, some large objects, etc. Such multi-level details are very important for recognizing driver behaviors. Second, PyConv is flexible and extensible, giving a large space of potential architecture designs. That is, PyConv gives strong potential to search for a lightweight architecture.

At the end of each block, we use a pooling layer to downsample the feature maps. Two types of pooling layers are widely used for this objective: max pooling and average pooling. We define our search space as the candidates of different designs of PyConv and different pooling types in the four convolutional blocks.

As shown in Figure 2, the overall process of defining the architecture of the student network is given as: at first, we construct a super student network covering all the candidates of each block. In the super student network, the output feature maps of the candidates of each block are aggregated by a weighted sum to become the input of the next block. The sum weights are learnable and represent the probability of choosing the candidates. Then the super student network is trained to learn from the teacher network. Thereafter, the final architecture of the student network is derived by selecting the candidate with the maximum probability.

Specifically, let $\{b_1, b_2, b_3, b_4\}$ denote the four blocks of the student network and super student network. $\{\omega_b^1(.), \omega_b^2(.), \ldots, \omega_b^m(.), \ldots, \omega_b^M(.)\}$ denote M different candidates of PyConv for the block b. $\{f_b^{\text{avg\_pool}}(.), f_b^{\text{max\_pool}}(.)\}$ denotes the candidates of using average pooling or max pooling layer at the end of the block b. Given the feature map X_b^{in} outputted by the previous block, the output feature map X_b^{out} of the block b in the super student network is defined as:

$$X_b^{\text{out}} = \beta_b^{\text{avg}} f_b^{\text{avg\_pool}}\Big(\sum_{m=1}^{M} \alpha_b^m \omega_b^m (X_b^{\text{in}})\Big) + \beta_b^{\text{max}} f_b^{\text{max\_pool}}\Big(\sum_{m=1}^{M} \alpha_b^m \omega_b^m (X_b^{\text{in}})\Big), \tag{8}$$

where $\{\alpha_b^1, \alpha_b^2, \ldots, \alpha_b^m, \ldots, \alpha_b^M\}$ and $\{\beta_b^{\text{avg}}, \beta_b^{\text{max}}\}$ are the probabilities of choosing the corresponding candidates, and they are computed as:

$$\alpha_b^m = \frac{\exp(\hat{\alpha}_b^m)}{\sum_{j=1}^{M} \exp(\hat{\alpha}_b^j)}, \tag{9}$$

$$\beta_b^g = \frac{\exp(\hat{\beta}_b^g)}{\exp(\hat{\beta}_b^{\text{avg}}) + \exp(\hat{\beta}_b^{\text{max}})}, \quad g = \{\text{avg}, \text{max}\}, \tag{10}$$

where, $\{\hat{\alpha}_b^1, \hat{\alpha}_b^2, \ldots, \hat{\alpha}_b^m, \ldots, \hat{\alpha}_b^M\}$ and $\{\hat{\beta_b}^{\text{avg}}, \hat{\beta_b}^{\text{max}}\}$ are learnable parameters that are all initialized as 1 and optimized during the training.

All the blocks of $\{b_1, b_2, b_3, b_4\}$ of the super student network are constructed by the process introduced above. The output feature map of b_4 is processed by the GAP and FC layers to predict the probability distribution over the classes (p_{super}). As mentioned above, after the training of the teacher network, we only use p_{N+1} of the teacher network for category prediction. The super student network is trained with the

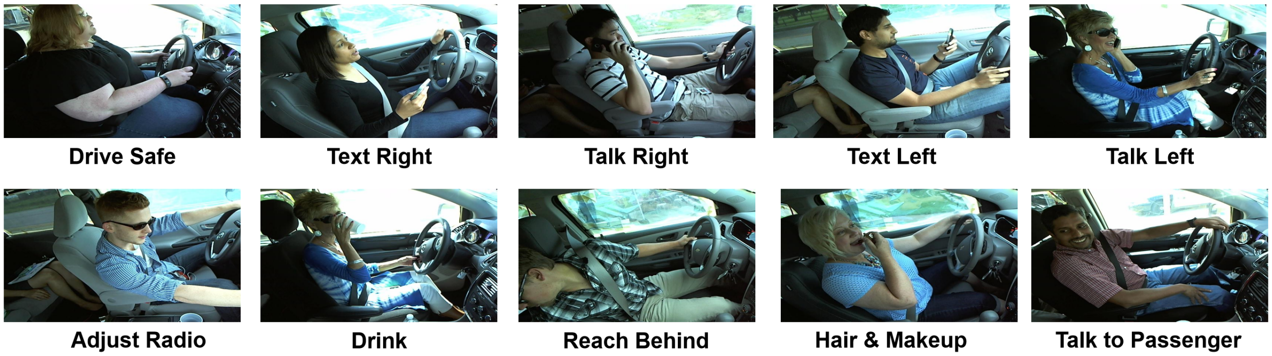


Fig. 3. Sample images of distracted driving behaviors on the Statefarm Distracted Driver Detection Dataset (SFD3) [46].

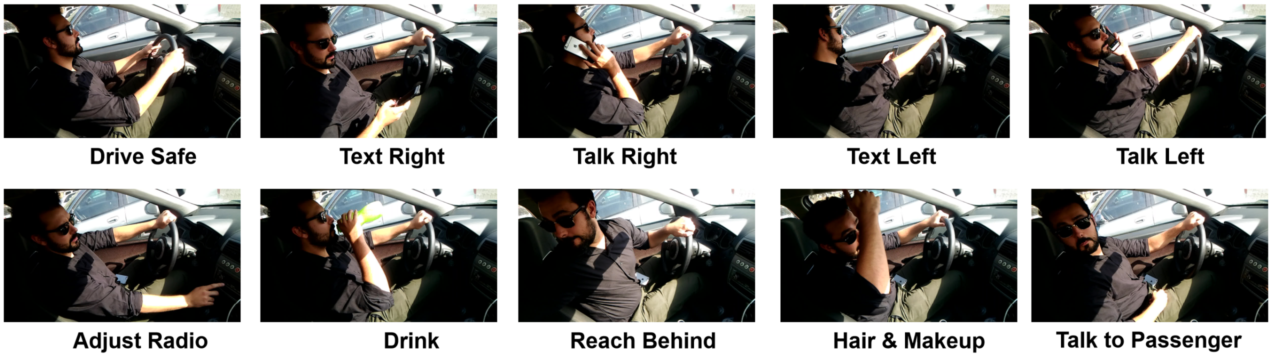


Fig. 4. Sample images of distracted driving behaviors on the AUC Distracted Driver Dataset (AUCD2) [24].

search loss $\mathcal{L}_{\text{search}}(.)$ defined as:

$$\begin{aligned}\mathcal{L}_{\text{search}}(p_{\text{super}}, p_{N+1}, p_{\text{truth}}) = {} & \lambda\mathcal{L}_{\text{mse}}(p_{\text{super}}, p_{N+1}) \\ & + (1-\lambda)\mathcal{L}_{\text{cls}}(p_{\text{super}}, p_{\text{truth}}),\end{aligned} \tag{11}$$

where $L_{\text{mse}}(.)$ denotes the mean squared error loss, and λ is a manual hyperparameter. During the training of the super student network, the parameters of the teacher network are fixed. After the training, we only keep the candidate with the maximum probability and prune all the other candidates for each block to construct the student network.

C. Knowledge Transfer

In the former subsection, we use the teacher network to guide the search of the student network architecture, and in this subsection, we use it to transfer knowledge to the student network. Assume that p_{student} is the probability distribution over the classes predicted by the student network. The student network is trained with the knowledge transfer loss $\mathcal{L}_{\text{trans}}(.)$ defined as:

$$\begin{aligned}\mathcal{L}_{\text{trans}}(p_{\text{student}}, p_{N+1}, p_{\text{truth}}) = {} & \lambda\mathcal{L}_{\text{mse}}(p_{\text{student}}, p_{N+1}) \\ & + (1-\lambda)\mathcal{L}_{\text{cls}}(p_{\text{student}}, p_{\text{truth}}).\end{aligned} \tag{12}$$

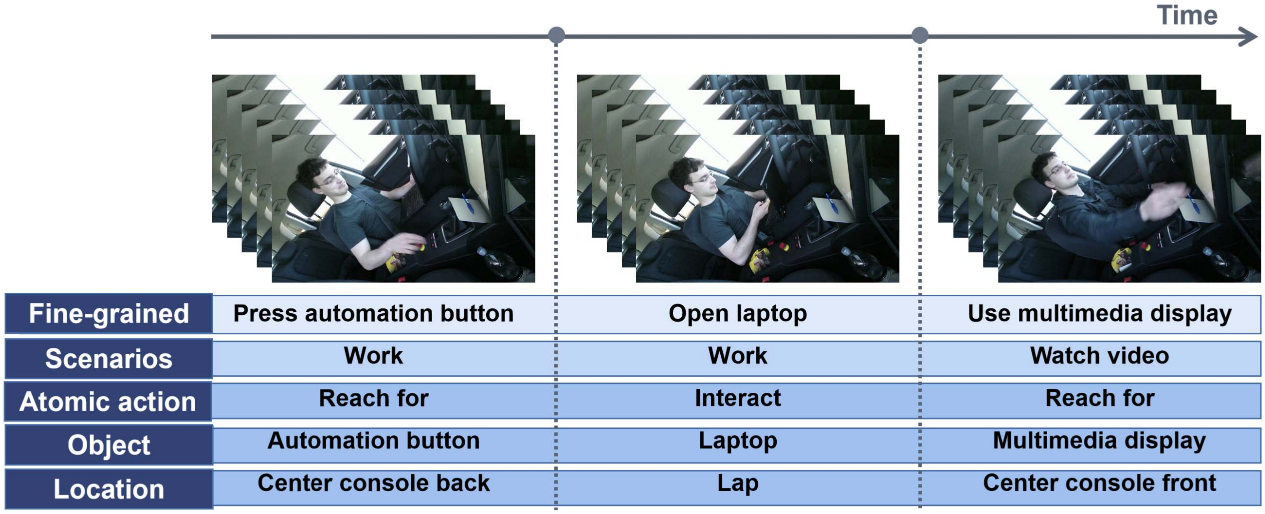
III. Datasets and Implementation Details

A. Dataset Description

The experiments are conducted using two types of datasets: image-based DDR dataset and video-based DDR dataset. The image-based DDR task requires recognizing the driver's behavior from each given image. The video-based DDR task requires recognizing the driver's behavior from each given video clip containing several frames. We mainly carried out the experiments of training the teacher network, defining the student network, and evaluating the performance of the teacher and student networks on the image-based DDR datasets. Then, we obtained an extremely lightweight yet powerful student network for the image-based DDR task. Thereafter, following Hara et al. [47], we extended the student network from 2D to 3D for the video-based DDR task.

For the image-based DDR task, we carried out experiments on two standard benchmark datasets for DDR: the Statefarm Distracted Driver Detection Dataset (SFD3) [46] and the AUC Distracted Driver Dataset (AUCD2) [24]. These two datasets are the most widely used datasets, and have been used for many studies on DDR. Both of the two datasets are composed of one safe driving action and nine distracted driving actions including (i) text right, (ii) talk right, (iii) text left, (iv) talk left, (v) adjust radio, (vi) drink, (vii) reach behind, (viii) hair and makeup, and (ix) talk to passenger. The images of both datasets are taken by dashboard cameras recording the driver's behavior. The sample images of the SFD3 and AUCD2 are shown in Figure 3 and Figure 4, respectively.

SFD3 is one of the most influential public datasets in the field of DDR. There are 22,424 images for training (around 2,000 images in each category) and 79,728 unlabeled images for testing. Since SFD3 does not provide the labels for the testing images, we follow the common practice of previous studies to perform experiments on the training dataset. We randomly split the training dataset of SFD3 as training image: testing image = 7:3 [13], [21], 7.5:2.5 [21], [54], [55], 8:2 [14],



Fine-grained	Press automation button	Open laptop	Use multimedia display
Scenarios	Work	Work	Watch video
Atomic action	Reach for	Interact	Reach for
Object	Automation button	Laptop	Multimedia display
Location	Center console back	Lap	Center console front

Fig. 5. Sample frames of distracted driving behaviors on the Drive&Act Dataset (DAD) [51]. The video clips are labeled with multiple annotations, including different categories of fine-grained activities, scenarios, atomic actions, objects, locations, together with possible combinations of the atomic actions, objects, and locations.

[21], [56], [57], [58], 9:1 [21], [59]. In this work, for each proportion of the train-test partition, we randomly split the images 10 times and report the average accuracy.

AUCD2 is another widely used public dataset for DDR. It has 17,308 RGB frames, of which 12,977 are for training, while the remaining 4,331 are for testing.

For the video-based DDR task, we utilized the Drive&Act Dataset (DAD) [51]. This is a large-scale video dataset consisting of various driver activities, with more than 9.6 million frames. As shown in Figure 5, the DAD provides multiple annotations for performing three types of recognition tasks on the video clips. The first task is the scenario recognition task, which requires recognizing the top-level activities (e.g., work and drink) from each given video clip. There are totally 12 different scenario categories. The second task is the fine-grained activity recognition task, which requires recognizing the specific semantic actions (e.g., open laptop, close bottle, etc.) from each video clip. There are totally 34 different categories of fine-grained activities. The third task is the atomic action unit recognition task. The atomic action units portray the lowest degree of abstraction and are basic driver interactions with the environment. The annotations of the atomic action units involve triplets of atomic action, object, and location, which are detached from long-term semantic meaning and can be regarded as building blocks for complex activities. There are five categories of atomic actions (e.g., reach for), 17 categories of objects (e.g., automation button), 14 categories of locations (e.g., center console back), and 372 possible combinations.

B. Implementation Details

1) Teacher Network: We use SKResNeXt50_32 × 4d [52] as the backbone of the feature extractor E. We divide SKResNeXt50_32 × 4d into three segments $\{m_1, m_2, m_3\}$. m_1 includes the Conv1–Conv3 of SKResNeXt50_32 × 4d. m_2 and m_3 respectively include the Conv4 and Conv5 of SKResNeXt50_32 × 4d.

2) Super Student Network: As mentioned above, we define our search space as the candidates of different designs for the four convolution blocks of the student network and construct a super student network to cover all the candidates. The specific candidates are shown in Table I. In Table I, the design of filters are illustrated by kernel size, number of channels, and number of groups. For example, $\begin{bmatrix} 11 \times 11, 16, 1 \\ 7 \times 7, 16, 1 \end{bmatrix}$ denotes a PyConv layer with two types of filters: one filter has 11 × 11 kernel size and the other has 7 × 7 kernel size. Both filters have 16 channels and 1 group. The pooling layers are illustrated by the type and window size. For example, "Avg. Pool $\begin{bmatrix} 2 \times 2 \end{bmatrix}$" denotes an average pooling layer with 2×2 window size. The stride of all the convolution layers is set as 1 and the padding size is set as $\frac{\theta - 1}{2}$, where θ is the spatial size of the filter. Thus, the convolution layers do not change the spatial size of feature maps. The stride of the pooling layers is set as 2, and the height and width of feature maps decrease by half after the pooling layers.

3) Training Details: For the experiments on the image-based datasets, during the training, all the learning rate are set as 0.002 with cosine annealing [60]. Weight decay is set as 5×10^{-4}. The input images are resized to 256 × 256 and applied with random crop of 224 × 224 region for training, center crop of 224 × 224 region for testing. We set batch size as 32 and train each network for 300 epochs. The manual hyper parameter λ in Equation 11 and Equation 12 is set as 0.7, which is a common setting for distillation. For the experiments on the video-based dataset, we follow the settings of Hara et al. [47]. Specifically, the learning rate is set as 0.001 with plateau scheduler [47]. Weight decay is set as 1×10^{-5}. The batch size is set as 32, and 16 frames (16 × 3 × 112 × 112) are sampled for each video clip by uniform sampling.

IV. Results and Discussions

A. Student Network Architecture Definition

As mentioned above, we first train the super student network to approximate the prediction distribution of the

TABLE I

CANDIDATES OF EACH BLOCK

Blocks	Block 1 (b_1)		Block 2 (b_2)		Block 3 (b_3)		Block 4 (b_4)	
Output Size	112×112, 32		56×56, 64		28×28, 128		14×14, 256	
Candidate 1	$\begin{bmatrix}11\times 11,16,1\\7\times 7,16,1\end{bmatrix}$	Avg. Pool $[2\times 2]$	$\begin{bmatrix}9\times 9,32,1\\5\times 5,32,1\end{bmatrix}$	Avg. Pool $[2\times 2]$	$\begin{bmatrix}5\times 5,64,1\\3\times 3,64,1\end{bmatrix}$	Avg. Pool $[2\times 2]$	$\begin{bmatrix}3\times 3,128,1\\1\times 1,128,1\end{bmatrix}$	Avg. Pool $[2\times 2]$
Candidate 2	$\begin{bmatrix}11\times 11,16,1\\5\times 5,16,1\end{bmatrix}$	Max Pool $[2\times 2]$	$\begin{bmatrix}9\times 9,32,1\\5\times 5,32,2\end{bmatrix}$	Max Pool $[2\times 2]$	$\begin{bmatrix}5\times 5,64,1\\3\times 3,64,2\end{bmatrix}$	Max Pool $[2\times 2]$	$\begin{bmatrix}3\times 3,128,1\\1\times 1,128,2\end{bmatrix}$	Max Pool $[2\times 2]$
Candidate 3	$\begin{bmatrix}11\times 11,16,1\\3\times 3,16,1\end{bmatrix}$	-	$\begin{bmatrix}9\times 9,16,1\\5\times 5,16,4\end{bmatrix}$	-	$\begin{bmatrix}5\times 5,64,1\\3\times 3,64,4\end{bmatrix}$	-	$\begin{bmatrix}3\times 3,128,1\\1\times 1,128,4\end{bmatrix}$	-
Candidate 4	$\begin{bmatrix}11\times 11,16,1\\1\times 1,16,1\end{bmatrix}$	-	$\begin{bmatrix}9\times 9,32,1\\3\times 3,32,1\end{bmatrix}$	-	$\begin{bmatrix}5\times 5,64,1\\1\times 1,64,1\end{bmatrix}$	-	-	-
Candidate 5	-	-	$\begin{bmatrix}9\times 9,32,1\\3\times 3,32,2\end{bmatrix}$	-	$\begin{bmatrix}5\times 5,64,1\\1\times 1,64,2\end{bmatrix}$	-	-	-
Candidate 6	-	-	$\begin{bmatrix}9\times 9,32,1\\3\times 3,32,4\end{bmatrix}$	-	$\begin{bmatrix}5\times 5,64,1\\1\times 1,64,4\end{bmatrix}$	-	-	-
Candidate 7	-	-	$\begin{bmatrix}9\times 9,32,1\\1\times 1,32,1\end{bmatrix}$	-	-	-	-	-
Candidate 8	-	-	$\begin{bmatrix}9\times 9,32,1\\1\times 1,32,2\end{bmatrix}$	-	-	-	-	-
Candidate 9	-	-	$\begin{bmatrix}9\times 9,32,1\\1\times 1,32,4\end{bmatrix}$	-	-	-	-	-

TABLE II

PROBABILITIES OF CHOOSING EACH CANDIDATE OF EACH BLOCK

	Block 1 (b_1)		Block 2 (b_2)		Block 3 (b_3)		Block 4 (b_4)	
Candidate 1	0.228	0.322	0.139	0.201	0.2074	0.043	0.966	0.052
Candidate 2	0.246	0.678	0.121	0.799	0.1865	0.957	0.018	0.948
Candidate 3	0.264	-	0.127	-	0.1622	-	0.016	-
Candidate 4	0.262	-	0.123	-	0.1504	-	-	-
Candidate 5	-	-	0.101	-	0.1507	-	-	-
Candidate 6	-	-	0.105	-	0.1428	-	-	-
Candidate 7	-	-	0.097	-	-	-	-	-
Candidate 8	-	-	0.096	-	-	-	-	-
Candidate 9	-	-	0.091	-	-	-	-	-

TABLE III

THE STUDENT NETWORK DEFINED BY DISTILLATION-BASED NEURAL ARCHITECTURE SEARCH

Block 1 (b_1)	Block 2 (b_2)	Block 3 (b_3)	Block 4 (b_4)	GAP $\mathscr{N}$-D FC Softmax
$\begin{bmatrix}11\times 11,16,1\\3\times 3,16,1\end{bmatrix}$ Batch Norm. ReLU Max Pool $[2\times 2]$	$\begin{bmatrix}9\times 9,32,1\\5\times 5,32,1\end{bmatrix}$ Batch Norm. ReLU Max Pool $[2\times 2]$	$\begin{bmatrix}5\times 5,64,1\\3\times 3,64,1\end{bmatrix}$ Batch Norm. ReLU Max Pool $[2\times 2]$	$\begin{bmatrix}3\times 3,128,1\\1\times 1,128,1\end{bmatrix}$ Batch Norm. ReLU Max Pool $[2\times 2]$	
112×112, 32	56×56, 64	28×28, 128	14×14, 256	1×1, $\mathscr{N}$

* $\mathscr{N}$ denotes the number of categories.

teacher network. We carry out this experiment on the AUCD2, as it is a more challenging dataset than SFD3. The probability of choosing each candidate is shown in Table II. The candidates of the highest probability are marked in gray background. For convolutional layers, the searching guided by the teacher network chooses the third candidate for b_1 and the first candidate for all the other blocks. For pooling layers, the second candidate (max pooling) is selected for all the blocks. The reason might be that max pooling selects the brighter pixels or the features corresponding to the sharp pixels, and therefore more robust to illumination changes.

Referring to Table I and Table II, we define the architecture of the student network as Table III. This architecture only requires 0.42M parameters. In the following experiments, we use this architecture as the student network on both datasets.

B. Recognition Performance of the Teacher and Student Network

In this subsection, we compare the recognition performance of the teacher network with and without progressive learning (PL), the student network trained from scratch and finetuned after transferring the teacher network's knowledge to the student network. The results are shown in Table IV.

On the AUCD2, PL improves the teacher network by 1.06%, which shows the effectiveness of PL. On the SFD3, the improvement brought by PL is small, which is 0.04%–0.11%. It is because backbone of the teacher has already achieved

TABLE IV

THE RECOGNITION ACCURACY OF THE TEACHER AND STUDENT NETWORKS

	Teacher Network		Student Network	
	Backbone	Backbone +PL	Train from Scratch	Finetune after Distillation
AUCD2	95.29%	96.35%	95.12%	95.64%
SFD3	Train:Test=7:3			
	99.75±0.12%	99.87±0.03%	99.81±0.09%	99.87±0.03%
	Train:Test=7.5:2.5			
	99.79±0.11%	99.88±0.04%	99.82±0.09%	99.88±0.04%
	Train:Test=8:2			
	99.82±0.09%	99.89±0.05%	99.87±0.07%	99.89±0.05%
	Train:Test=9:1			
	99.87±0.05%	99.91±0.05%	99.87±0.05%	99.91±0.05%

* For SFD3, we illustrate the accuracy range of 10 random splits.

TABLE V

THE F1-SCORE OF THE TEACHER AND STUDENT NETWORKS

	Teacher Network		Student Network	
	Backbone	Backbone +PL	Train from Scratch	Finetune after Distillation
Safe driving	93.60%	95.14%	93.43%	94.07%
Text Right	95.24%	96.44%	95.13%	95.73%
Talk Right	94.74%	96.21%	94.25%	94.93%
Text left	95.09%	96.04%	95.09%	95.40%
Talk left	95.86%	97.35%	95.42%	95.86%
Adjust Radio	95.58%	96.56%	95.42%	95.60%
Drink	96.54%	97.29%	96.30%	97.04%
Reach Behind	95.41%	96.30%	95.41%	95.76%
Hair & Makeup	95.64%	96.48%	95.64%	96.48%
Talk to Passenger	96.68%	97.00%	96.60%	96.84%
Average	95.44%	96.48%	95.27%	95.77%

TABLE VI

COMPARISON OF THE DESIGNED STUDENT NETWORKS WITH EXISTING LIGHTWEIGHT NETWORKS IN TERMS OF GFLOPS AND TIME CONSUMPTION

		GFLOPs	Time Cost	
			Single Image	A Batch of 32 Images
Image	MobileVGG [20]	2.11	5.19ms	122.34ms
	MobileNet [17]	0.59	3.54ms	53.15ms
	MobileNetV2 [18]	0.33	6.94ms	44.72ms
	SqueezeNet [19]	0.86	3.86ms	48.79ms
	D-HCNN [21]	31.10	7.40ms	40.67ms
	2D Student Network	**2.25**	**2.23ms**	**35.69ms**
			Single Video Clip	A Batch of 8 Clips
Video Clip	C3D [48]	38.55	25.57ms	1452.86ms
	P3D ResNet [49]	18.67	148.04ms	983.22ms
	I3D [50]	27.90	61.86ms	588.44ms
	3D Student Network	**37.20**	**25.35ms**	**479.83ms**

a high accuracy that is 99.75%–99.87%. Considering the very narrow possible improvement space, we suppose PL can be still regarded as effective on the SFD3. In the following experiments, we use the teacher network with PL to guide the search of the student network architecture and transfer knowledge to the student network.

On both datasets, the student network trained from scratch already achieves a very high accuracy, which shows the architecture obtained by the proposed searching approach is effective for the DDR task. Knowledge distillation respectively improve 0.52% and 0.03%–0.05% on the AUCD2 and SFD3, respectively.

Considering that the accuracy for the datasets is almost saturated, it is interesting to see there is still room for the improvement by our proposed method.

In addition, since the AUCD2 dataset is somewhat unbalanced, we also show the F1-score obtained with this dataset in Table V. PL improves the teacher network by 0.32%–1.54% in different categories. Knowledge distillation respectively improve 0.17%–0.84% for the student network in different categories.

C. Comparison With State-of-the-Art Distracted Driver Recognition Approaches

In this subsection, we compare our performance with the state-of-the-art approaches on AUCD2 and SFD3.

Table VII shows the results on the AUCD3. The accuracy of the teacher network (96.35%) surpasses the best previous accuracy (96.31%), which is achieved by Regularized VGG-16 [12]. Regularized VGG-16 has 140M parameters, whereas the teacher network in this work has 44.62M parameters (i.e., 31.87% of the Regularized VGG-16 parameters), which shows the effectiveness of the teacher network on this dataset. The student network achieves 95.64% with 0.42M parameters. For comparison, the original VGG-16 achieves 94.44% with 140M parameters (i.e., 333.33 times of the student network parameters), and the modified VGG-16 achieves 96.54% with 15M parameters (i.e., 35.71 times of the student network parameters) [12].

Table VIII shows the results on the SFD3. Both the teacher and student network achieve 99.86%–99.91%, which outperforms the best previous accuracy. The student network is recommended because it requires fewer parameters.

D-HCNN [21] also achieves good accuracy on both datasets with small parameters. However, our student network is better because: (i) The student network has better accuracy than D-HCNN on both datasets; (ii) The student network's parameters are only about 55.26% of D-HCNN; (iii) D-HCNN requires HOG images in addition to RGB images as input. Therefore, it needs to compute the HOG feature [22] of every image when using D-HCNN, which is unfavorable for real-world applications.

Moreover, the student network has better real-time performance than other lightweight models. As shown in Table VI, for processing a single image in the test mode, the student

TABLE VII

COMPARISON WITH STATE-OF-THE-ART METHODS ON AUCD2

Approach	**Parameter Size**	**Accuracy**
AlexNet on Original Scene [8]	62M	93.65%
AlexNet on Skin Segmentation [8]	62M	93.60%
AlexNet on Face Segmentation [8]	62M	86.68%
AlexNet on Hand Segmentation [8]	62M	89.52%
AlexNet on Face + Hand Segmentation [8]	62M	86.68%
AlexNet on Original Scene [8]	24M	95.17%
AlexNet on Skin Segmentation [8]	24M	94.57%
AlexNet on Face Segmentation [8]	24M	88.82%
AlexNet on Hand Segmentation [8]	24M	91.62%
AlexNet on Face + Hand Segmentation [8]	24M	90.88%
Majority Voting Ensemble [8]	120M	95.77%
GA Weighted Ensemble [8]	120M	95.98%
Original VGG-16 [12]	140M	94.44%
Regularized VGG-16 [12]	140M	96.31%
Modified VGG-16 [12]	15M	95.54%
Pose-guided DenseNet [61]	8.06M	94.20%
MobileNet [17]	4.20M	94.67%
MobileNetV2 [18]	3.50M	94.74%
NasNet Mobile [33]	5.30M	94.69%
SqueezeNet [19]	1.25M	93.21%
MobileVGG [20]	2.20M	95.24%
VGG-one-attention [10]	>140M	84.82%
VGG-two-way-attention [10]	>140M	87.74%
D-HCNN [21]	0.76M	95.59%
Teacher Network (Ours)	**44.62M**	**96.35%**
Student Network (Ours)	**0.42M**	**95.64%**

network requires 2.25 GFLOPs takes 2.23 ms on 1080Ti + Intel i7-10700F. In comparison, MobileVGG [20] requires 2.11 GFLOPs and takes 5.19 ms. MobileNet [17] requires 0.59 GFLOPs and takes 3.54 ms. MobileNetV2 [18] requires 0.33 GFLOPs and takes 6.94 ms. SqueezeNet [19] requires 0.86 GFLOPs and takes 3.86 ms. D-HCNN [21] requires 31.10 GFLOPs and takes 7.40 ms. As D-HCNN requires HOG images as additional input, it takes additional 1.48ms per image to compute HOG for each image. Compared to previous lightweight networks, our network has no significant advantage in terms of GFLOPs but clearly has faster speed. It is because the parallelism of a convolutional network is mainly reflected in the calculation of each layer, and there is generally no parallelism across layers. So for convolutional neural networks used in high-speed DDR, large convolutional filter size is better than too deep layers. This fact was also pointed out by Qin et al. [21] and experimentally proved by them. Another advantage of our network is the aforementioned lower number of parameters, which allows our network to require less storage and memory space and be more easily deployed on in-vehicle devices.

Since GPUs can process multiple images in parallel, we also compare the time consumption of our network with other lightweight networks that process multiple images in parallel. For processing one batch of images (32 images) in the test mode, the student network takes 35.69 ms on 1080Ti + Intel i7-10700F. In comparison, MobileVGG [20] takes 122.34 ms. MobileNet [17] takes 53.15 ms. MobileNetV2 [18] takes 44.72 ms. SqueezeNet [19] takes 48.79 ms. D-HCNN [21] takes 40.67 ms.

As the proposed teacher and student networks achieve very high accuracy on both image-based DDR datasets [24], [46], it is important to know what images cause the small number of recognition failures. Figure 6 shows the typical failure cases of the wrongly-predicted images together with their ground-truth

TABLE VIII

COMPARISON WITH STATE-OF-THE-ART METHODS ON SFD3

Approach	**Parameter Size**	**Accuracy**
Train:Test=7:3		
VGG-16 [13]	140M	58.3%
VGG-19 [13]	142M	55.7%
Inception-V3 [13]	25.6M	92.90%
Inception-V3+Xception [13]	22.9M	82.50%
Inception-V3+Xception +ResNet50+VGG-19 [13]	46.7M	90.00%
D-HCNN [21]	0.76M	99.82%
Teacher Network (Ours)	**44.62M**	**99.87±0.03%**
Student Network (Ours)	**0.42M**	**99.87±0.03%**
Train:Test=7.5:2.5		
Unpretrained VGG-16 [54]	140M	99.43%
Pretrained VGG-16 [54]	140M	99.57%
Unpretrained VGG-19 [54]	142M	98.98%
Pretrained VGG-19 [54]	142M	99.39%
MobileVGG [20]	2.2M	99.75%
Transfer Learning With ResNet [55]	60M	85.00%
D-HCNN [21]	0.76M	99.84%
Teacher Network (Ours)	**44.62M**	**99.88±0.04%**
Student Network (Ours)	**0.42M**	**99.88±0.04%**
Train:Test=8:2		
Pixel SVC [14]	-	18.3%
SVC+HOG [14]	-	28.2%
SVC+PCA [14]	-	34.8%
SVC+Bbox+PCA [14]	-	40.7%
Original VGG-16 [14]	140M	90.2%
VGG-GAP [14]	140M	91.3%
Original VGG-16+VGG-GAP [14]	280M	92.6%
MLP [56]	-	82.00%
RNN [56]	-	91.7%
Drive-Net [56]	-	95.00%
Vanilla CNN with Data Transfer Learning [57]	26.05M	97.05%
CNN with Data Transfer Learning [57]	3.5M	71.72%
HCF [58]	>72.3M	96.74%
D-HCNN [21]	0.76M	99.86%
Teacher Network (Ours)	**44.62M**	**99.89±0.05%**
Student Network (Ours)	**0.42M**	**99.89±0.05%**
Train:Test=9:1		
AlexNet+SoftmaxLoss [59]	63.2M	96.80%
AlexNet+TripletLoss [59]	63.2M	98.70%
D-HCNN [21]	0.76M	99.87%
Teacher Network (Ours)	**44.62M**	**99.91±0.05%**
Student Network (Ours)	**0.42M**	**99.91±0.05%**

* We illustrate the accuracy range of 10 random splits.

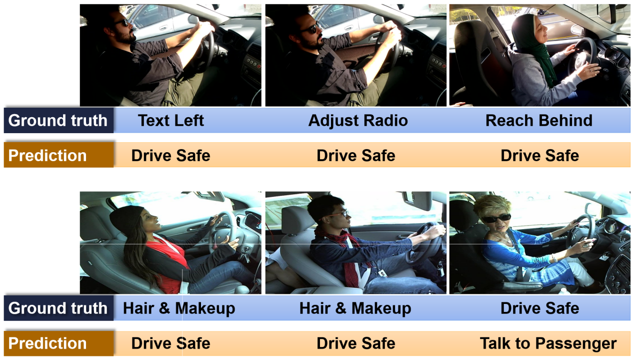


Fig. 6. Typical sample images that are wrongly classified by the networks proposed in this work. The ground-truth labels are confusing even for humans.

TABLE IX

THE RECOGNITION ACCURACY OF THE 3D STUDENT NETWORK ON THE THREE SPLITS OF DIFFERENT TASKS OF THE DAD [51]

	Fine-grained Activities		Scenarios		Atomic Action Units							
					Action		Object		Location		Action-Object-Location	
	Validation	Test	Validation	Test	Validation	Test	Validation	Test	Validation	Test	Validation	Test
Split 1	72.04%	69.57%	60.24%	44.97%	79.25%	84.81%	71.59%	65.50%	69.90%	63.79%	44.89%	51.22%
Split 2	67.75%	66.08%	46.04%	47.68%	81.16%	79.77%	61.13%	63.54%	66.48%	61.27%	47.64%	36.69%
Split 3	71.59%	61.43%	58.57%	35.80%	81.78%	74.70%	66.22%	46.96%	50.21%	65.37%	41.15%	41.08%
Average	70.46%	65.69%	54.95%	42.82%	80.73%	79.76%	66.31%	58.67%	62.20%	63.48%	44.56%	43.00%

TABLE X

COMPARISON WITH STATE-OF-THE-ART METHODS ON THE DAD [51]

		Parameter Size	Accuracy											
			Fine-grained Activities		Scenarios		Atomic Action Units							
							Atomic Action		Object		Location		Action-Object -Location	
			Val	Test	Val	Test	Val	Test	Val	Test	Val	Test	Val	Test
Body Pose Representation	Interior [51]	-	45.23%	40.30%	35.76%	29.75%	57.62%	47.74%	51.45%	41.72%	53.31%	52.64%	9.18%	7.07%
	Pose [51]	-	53.17%	44.36%	37.18%	32.96%	54.23%	49.03%	49.90%	40.73%	53.76%	53.33%	8.76%	6.85%
	Two-Stream [62]	-	53.76%	45.39%	39.37%	34.81%	57.86%	48.83%	52.72%	42.79%	53.99%	54.73%	10.31%	7.11%
	Three-Stream [63]	-	55.67%	46.95%	41.70%	35.45%	59.29%	50.65%	55.59%	45.25%	59.54%	56.50%	11.57%	8.09%
End-to-end	C3D [48]	78.14M	49.54%	43.41%	-	-	-	-	-	-	-	-	-	-
	P3D ResNet [49]	65.74M	55.04%	45.32%	-	-	-	-	-	-	-	-	-	-
	I3D [50]	12.32M	69.57%	63.64%	44.66%	31.80%	62.81%	56.07%	61.81%	56.15%	47.70%	51.12%	15.56%	12.12%
3D Student Network (Ours)		**2.03M**	**70.46%**	**65.69%**	**54.95%**	**42.82%**	**80.73%**	**79.76%**	**66.31%**	**58.67%**	**62.20%**	**63.48%**	**44.56%**	**43.00%**

labels and the prediction given by the proposed networks (the teacher network or student network). Those failure cases are even confusing for humans.

D. Extending the Student Network to 3D for the Video-Based Distracted Driver Recognition

The above experiments have proposed a lightweight yet powerful network architecture (i.e., the student network) for image-based DDR. In this subsection, we extend the student network into a spatial-temporal 3D network to evaluate whether on the video-based DDR dataset [51], the 3D student network can retrace the success of the student network architecture proposed for the image-based DDR. This experiment is inspired by the experiments of Hara et al. [47], in which the researchers replaced the 2D layers (e.g., 2D convolutional layers, 2D batch normalization layers, etc.) of the ResNet architectures [64] with 3D layers (e.g., 3D convolutional layers, 3D batch normalization layers, etc.) and proved that using 3D ResNet architectures together with Kinetics [65] can retrace the successful history of 2D CNNs on ImageNet [66]. Following Hara et al. [47], we set the size of the third dimension of each 3D convolutional kernel to be the same as the size of the first and second dimensions. For example, a 2D convolutional kernel of a 3 × 3 kernel size is extended to a 3D convolutional kernel of a 3 × 3 × 3 kernel size.

We conducted comprehensive experiments to evaluate the performance of the 3D student network for all the tasks on the DAD. The specific accuracy of each split and the average accuracy over the three splits are shown in Table IX. The comparison results with the state-of-the-art approaches on the DAD are shown in Table X. It can be observed that the 3D student network outperforms the state-of-the-art approaches by a significantly large margin in both validation and testing sets. Our approach is 0.89%–29.00% higher than the previous best accuracy in the validation set and 2.05%–30.88% higher than the previous best accuracy in the test set. Besides, the 3D student network has only 2.03M parameters and is much more lightweight than the state-of-the-art approaches. The parameter size of the 3D student network is only 16.48% of the parameter size of C3D [48], 3.09% of the parameter size of P3D ResNet [49], 2.60% of the parameter size of I3D [50]. Moreover, the student network has better real-time performance than those 3D convolutional neural networks. As shown in Table VI, for processing a single video clips, 3D student network requires 37.20 GFLOPs and takes 25.35 ms on 1080Ti + Intel i7-10700F. In comparison, C3D requires 38.55 GFLOPs and takes 25.57 ms. P3D ResNet requires 18.67 GFLOPs and takes 148.04 ms. I3D 27.90 GFLOPs and takes 61.86 ms. Similar to the case of 2D student network, 3D student network does not have clear advantage in terms of GFLOPs, but has a clearly faster speed.

For processing 8 video clips (8 × 16 frames) in the test mode, the 3D student network takes 479.83 ms on 1080Ti + Intel i7-10700F. In comparison, C3D takes 1452.86 ms. P3D ResNet takes 983.22 ms. I3D takes 588.44 ms.

E. Discussion on the Implication of the Proposed Framework on the Its Applications

The implication of our approach to applications is as follows:

- We construct a powerful teacher network using progressive learning to increase robustness to illumination changes from shallow to deep layers of a backbone CNN. The classification accuracy of the teacher network exceeds that of all existing approaches and is well suited for the DDR applications that do not require a particularly small computational overhead but rather high accuracy.

TABLE XI
THE RECOGNITION ACCURACY OF THE TEACHER AND STUDENT NETWORK ON THE THREE ADDITIONAL DATASETS

	Teacher Network		Student Network	
	Backbone	Backbone +PL	Train from Scratch	Finetune after Distillation
SLD2 [67]	99.53%	99.74%	99.48%	99.74%
Gesture2012 [68]	100%	100%	100%	100%
USED [69]	97.91%	98.75%	89.17%	92.08%

TABLE XII
COMPARISON RESULTS ON THE THREE ADDITIONAL DATASETS

Approach	**Parameter Size**	**Accuracy**
SLD2 [67]		
Contour SVM-based digit-gesture recognition	-	69.00%
CNN-based digit-gesture recognition [70]	1.84M	98.32%
Increasing Filter Size	5.46M	99.68%
Decreasing Filter Size [21]	0.76M	99.68%
Teacher Network (Ours)	**44.62M**	**99.74%**
Student Network (Ours)	**0.42M**	**99.74%**
Gesture2012 [68]		
CNN-based digit-gesture recognition [70]	1.84M	100%
Increasing Filter Size	5.46M	96.30%
Decreasing Filter Size [21]	0.76M	94.10%
Teacher Network (Ours)	**44.62M**	**100%**
Student Network (Ours)	**0.42M**	**100%**
USED [69]		
SIFT+GGM [69]	-	73.4%
HMP	-	85.7%
SIFT+SC [71]	-	82.7%
OB [72]	-	76.3%
Places-CNN	61M	94.12%
ImageNet-CNN	61M	94.42%
Hybrid-CNN [73]	122M	94.22%
TPN-FS [74]	121M	95.2%
DTCTH(LSVM)	-	85.16%
DTCTH(HI) [75]	-	88.18%
CLGC(RGB-RGB)	-	86.4%
CLGC(RGB-HSV) [76]	-	90.85%
Teacher Network (Ours)	**44.62M**	**98.75%**
Student Network (Ours)	**0.42M**	**92.08%**

- Using NAS and knowledge distillation, we generate an effective student network with the guidance of the teacher network. The student network can achieve high DDR accuracy and has less parametric count and inference time than any existing lightweight DDR networks. The student network is suitable for applications with high parametric and inference time requirements.
- We extend the student network into a spatial-temporal 3D network for performing DDR based on small video clips. The 3D student network has better DDR accuracy, smaller parameter size, and faster speed than the existing approaches. The 3D student network is suitable for applications developed based on video clips.
- Our proposed framework combining knowledge distillation and NAS has the potential to become a general DDR network design framework for different applications.

V. ADDITIONAL EXPERIMENTS

We also evaluate our approach on three additional datasets, which are not for the DDR task but have the same characteristic: small diversity and strong inter-class similarity. The three additional datasets are Sign Language Digits Dataset (SLD2) [67], Gesture Dataset 2012 (Gesture2012) [68], and UIUC Sports Event Dataset (USED) [69]. SLD2 and Gesture2012 are image datasets for hand sign language recognition, which are also used by Qin et al. [21] as additional datasets to evaluate D-HCNN [21]. USED is an image dataset for sport event recognition.

On the three additional datasets, we compared the recognition performance of the teacher network with and without progressive learning (PL), the student network trained from scratch and finetuned after knowledge transferring. The results are shown in Table XI. We also compared our approach with the state-of-the-art approaches on the three additional datasets, and the results are shown in the Table XII.

Both the teacher and student networks achieve 99.74% on the SLD and 100% on the Gesture2012, which reach state-of-the-art performance on the two datasets. The student network has much fewer parameters than other state-of-the-art approaches on these two datasets.

On the USED, the improvement brought by PL and knowledge transfer is obvious. PL improves the teacher network by 0.84% and knowledge transfer improves the student network by 2.91%. The accuracy of the teacher network is 98.75%, which surpasses the best previous accuracy by 3.55%. The student network achieves 92.08% with 0.42M parameters.

VI. CONCLUSION

In this paper, we proposed a novel framework for distracted driver recognition to achieve high accuracy with a small number of parameters. This framework first builds a powerful teacher network based on progressive learning and then uses the teacher network to guide the searching of an optimal architecture for a student network, which is lightweight but can achieve high accuracy. Thereafter, the teacher network is used again to transfer the knowledge to the student network. The teacher network outperforms the previous state-of-the-art approaches on the Statefarm Distracted Driver Detection Dataset and AUC Distracted Driver Dataset. The student network achieves high accuracy with extremely tiny parameters on both datasets. The student network architecture can be extended into a spatial-temporal 3D convolutional neural network for recognizing distracted driving behaviors from video clips. The 3D student network significantly outperforms the previous state-of-the-art approaches with only 2.03M parameters on the Drive&Act Dataset.

REFERENCES

[1] NHTSA, "2015 motor vehicle crashes: Overview," *Traffic Saf. Facts, Res. Note*, vol. 2016, pp. 1–9, Aug. 2016.

[2] S. N. Resalat and V. Saba, "A practical method for driver sleepiness detection by processing the EEG signals stimulated with external flickering light," *Signal, Image Video Process.*, vol. 9, no. 8, pp. 1751–1757, Nov. 2015.

[3] NHTSA, "Overview of the 2019 crash investigation sampling system," *Traffic Saf. Facts, Res. Note*, vol. 2020, pp. 1–14, Dec. 2020.

[4] S. M. Iranmanesh, H. N. Mahjoub, H. Kazemi, and Y. P. Fallah, "An adaptive forward collision warning framework design based on driver distraction," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 12, pp. 3925–3934, Dec. 2018.

[5] S. M. Petermeijer, J. C. F. de Winter, and K. J. Bengler, "Vibrotactile displays: A survey with a view on highly automated driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 897–907, Apr. 2015.

[6] F. Vicente, Z. Huang, X. Xiong, F. D. L. Torre, W. Zhang, and D. Levi, "Driver gaze tracking and eyes off the road detection system," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 2014–2027, Aug. 2015.

[7] A. Nemcova et al., "Multimodal features for detection of driver stress and fatigue," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3214–3233, Jun. 2021.

[8] Y. Abouelnaga, H. M. Eraqi, and M. N. Moustafa, "Real-time distracted driver posture classification," in *Proc. NIPS Workshop Mach. Learn. Intell. Transp. Syst.*, 2018, pp. 1–8.

[9] M. Wollmer et al., "Online driver distraction detection using long short-term memory," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 574–582, Jun. 2011.

[10] Y. Ai, J. Xia, K. She, and Q. Long, "Double attention convolutional neural network for driver action recognition," in *Proc. 3rd Int. Conf. Electron. Inf. Technol. Comput. Eng.*, 2019, pp. 1515–1519.

[11] R. O. Mbouna, S. G. Kong, and M.-G. Chun, "Visual analysis of eye state and head pose for driver alertness monitoring," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1462–1469, Sep. 2013.

[12] B. Baheti, S. Gajre, and S. Talbar, "Detection of distracted driver using convolutional neural network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 1032–1038.

[13] K. R. Dhakate and R. Dash, "Distracted driver detection using stacking ensemble," in *Proc. IEEE Int. Students' Conf. Elect., Electron. Comput. Sci. (SCEECS)*, Feb. 2020, pp. 1–5.

[14] B. Zhang, "Apply and compare different classical image classification method: Detect distracted driver," Stanford Univ., Stanford, CA, USA, Project Rep. CS 229, 2016.

[15] A. Kashevnik, R. Shchedrin, C. Kaiser, and A. Stocker, "Driver distraction detection methods: A literature review and framework," *IEEE Access*, vol. 9, pp. 60063–60076, 2021.

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

[17] A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.

[18] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Dec. 2018, pp. 4510–4520.

[19] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: Alexnet-level accuracy with 50x fewer parameters and <0.5 MB model size," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–13.

[20] B. Baheti, S. Talbar, and S. Gajre, "Towards computationally efficient and realtime distracted driver detection with MobileVGG network," *IEEE Trans. Intell. Vehicles*, vol. 5, no. 4, pp. 565–574, Dec. 2020.

[21] B. Qin, J. Qian, Y. Xin, B. Liu, and Y. Dong, "Distracted driver detection based on a CNN with decreasing filter size," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 6922–6933, Jul. 2022.

[22] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, no. 1, pp. 886–893, Jun. 2005.

[23] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *Int. J. Comput. Vis.*, vol. 129, pp. 1789–1819, Mar. 2021.

[24] M. Alotaibi and B. Alotaibi, "Distracted driver classification using deep learning," *Signal, Image Video Process.*, vol. 14, no. 3, 2019, pp. 1–8.

[25] R. Du et al., "Fine-grained visual classification via progressive multi-granularity training of jigsaw patches," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 153–168.

[26] P. Zhao, Q. Miao, H. Yao, X. Liu, R. Liu, and M. Gong, "CA-PMG: Channel attention and progressive multi-granularity training network for fine-grained visual classification," *IET Image Process.*, vol. 15, no. 14, pp. 3718–3727, Dec. 2021.

[27] N. Ahn, B. Kang, and K.-A. Sohn, "Image super-resolution via progressive cascading residual network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2018, pp. 791–799.

[28] Y. Wang, F. Perazzi, B. McWilliams, A. Sorkine-Hornung, O. Sorkine-Hornung, and C. Schroers, "A fully progressive approach to single-image super-resolution," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 864–873.

[29] T. R. Shaham, T. Dekel, and T. Michaeli, "SinGAN: Learning a generative model from a single natural image," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 4570–4580.

[30] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 818–833.

[31] A. B. Jung et al., *Imgaug*. Accessed: Feb. 1, 2020. [Online]. Available: https://github.com/aleju/imgaug

[32] T. Elsken, J. H. Metzen, and F. Hutter, "Neural architecture search: A survey," *J. Mach. Learn. Res.*, vol. 20, no. 1, pp. 1997–2017, 2019.

[33] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710.

[34] P. Bashivan, M. Tensen, and J. Dicarlo, "Teacher guided architecture search," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5320–5329.

[35] C. Liu et al., "Progressive neural architecture search," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 19–34.

[36] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameters sharing," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4095–4104.

[37] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, 2019, pp. 4780–4789.

[38] L. Hanxiao, S. Karen, and Y. Yiming, "DARTS: Differentiable architecture search," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–13.

[39] K. Kirthevasan, N. Willie, P. Jeff, S. Barnabás, and P. X. Eric, "Neural architecture search with Bayesian optimisation and optimal transport," in *Proc. Conf. Neural Inf. Process. Syst.*, 2018, pp. 1–10.

[40] X. Wang, "Teacher guided neural architecture search for face recognition," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, no. 4, 2021, pp. 2817–2825.

[41] R. Luo, F. Tian, T. Qin, E. Chen, and T.-Y. Liu, "Neural architecture optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Red Hook, NY, USA: Curran, 2018, pp. 1–12.

[42] P. Welinder et al., "Caltech-UCSD Birds 200," California Inst. Technol., Pasadena, CA, USA, Tech. Rep. CNS-TR-2010-001, 2010.

[43] I. C. Duta, L. Liu, F. Zhu, and L. Shao, "Pyramidal convolution: Rethinking convolutional neural networks for visual recognition," 2020, *arXiv:2006.11538*.

[44] D. Yu, H. Wang, P. Chen, and Z. Wei, "Mixed pooling for convolutional neural networks," in *Proc. Int. Conf. Rough Sets Knowl. Technol.*, 2014, pp. 364–375.

[45] J. Chen et al., "Fine-grained detection of driver distraction based on neural architecture search," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 9, pp. 5783–5801, Sep. 2021.

[46] StateFarm. *State Farm Distracted Driver Detection*. Accessed: Jun. 15, 2017. [Online]. Available: https://www.kaggle.com/c/state-farm-distracted-driver-detection

[47] K. Hara, H. Kataoka, and Y. Satoh, "Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and imagenet?" in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6546–6555.

[48] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.

[49] Z. Qiu, T. Yao, and T. Mei, "Learning spatio-temporal representation with pseudo-3D residual networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5533–5541.

[50] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6299–6308.

[51] M. Martin et al., "Drive&Act: A multi-modal dataset for fine-grained driver behavior recognition in autonomous vehicles," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, Oct. 2019, pp. 2801–2810.

[52] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 510–519.

[53] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[54] S. Masood, A. Rai, A. Aggarwal, M. N. Doja, and M. Ahmad, "Detecting distraction of drivers using convolutional neural network," *Pattern Recognit. Lett.*, vol. 139, pp. 79–85, Nov. 2020.

[55] M. D. Hssayeni, S. Saxena, R. Ptucha, and A. Savakis, "Distracted driver detection: Deep learning vs handcrafted features," *Electron. Imag.*, vol. 29, no. 10, pp. 20–26, Jan. 2017.

[56] M. S. Majdi, S. Ram, J. T. Gill, and J. J. Rodriguez, "Drive-Net: Convolutional network for driver distraction detection," in *Proc. IEEE Southwest Symp. Image Anal. Interpretation (SSIAI)*, Apr. 2018, pp. 1–4.

[57] A. Jamsheed, V. B. Janet, and U. S. Reddy, "Real time detection of driver distraction using CNN," in *Proc. 3rd Int. Conf. Smart Syst. Inventive Technol. (ICSSIT)*, Aug. 2020, pp. 185–191.

[58] C. Huang, X. Wang, J. Cao, S. Wang, and Y. Zhang, "HCF: A hybrid CNN framework for behavior detection of distracted drivers," *IEEE Access*, vol. 8, pp. 109335–109349, 2020.

[59] O. D. Okon and L. Meng, "Detecting distracted driving with deep learning," in *Proc. Int. Conf. Interact. Collaborative Robot.*, 2017, pp. 170–179.

[60] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–16.

[61] A. Behera and A. H. Keidel, "Latent body-pose guided DenseNet for recognizing driver's fine-grained secondary activities," in *Proc. 15th IEEE Int. Conf. Adv. Video Signal Based Surveill. (AVSS)*, Nov. 2018, pp. 1–6.

[62] H. Wang and L. Wang, "Modeling temporal dynamics and spatial configurations of actions using two-stream recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 499–508.

[63] M. Martin, J. Popp, M. Anneken, M. Voit, and R. Stiefelhagen, "Body pose and context information for driver secondary task detection," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2018, pp. 2015–2021.

[64] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[65] W. Kay et al., "The kinetics human action video dataset," 2017, *arXiv:1705.06950*.

[66] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[67] A. Mavi, "A new dataset and proposed convolutional neural network architecture for classification of American sign language digits," 2020, *arXiv:2011.08927*.

[68] A. Barczak, N. Reyes, M. Abastillas, A. Piccio, and T. Susnjak, "A new 2D static hand gesture colour image dataset for ASL gestures," *Res. Lett. Inf. Math. Sci.*, vol. 15, 2011, pp. 12–20.

[69] L.-J. Li and L. Fei-Fei, "What, where and who? Classifying events by scene and object recognition," in *Proc. IEEE 11th Int. Conf. Comput. Vis.*, Oct. 2007, pp. 1–8.

[70] C. Saha, R. H. Faisal, and M. M. Rahman, "Bangla handwritten digit recognition using an improved deep convolutional neural network architecture," in *Proc. Int. Conf. Electr., Comput. Commun. Eng. (ECCE)*, Feb. 2019, pp. 1–6.

[71] B. Liefeng, R. Xiaofeng, and F. Dieter, "Hierarchical matching pursuit for image classification: Architecture and fast algorithms," in *Proc. Conf. Neural Inf. Process. Syst.*, 2011, pp. 1–10.

[72] L.-J. Li, H. Su, Y. Lim, and L. Fei-Fei, "Object bank: An object-level image representation for high-level visual recognition," *Int. J. Comput. Vis.*, vol. 107, no. 1, pp. 20–39, Mar. 2014.

[73] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 1–9.

[74] S. Bai, Z. Li, and J. Hou, "Learning two-pathway convolutional neural networks for categorizing scene images," *Multimedia Tools Appl.*, vol. 76, no. 15, pp. 16145–16162, 2017.

[75] M. M. Rahman, S. Rahman, R. Rahman, B. M. M. Hossain, and M. Shoyaib, "DTCTH: A discriminative local pattern descriptor for image classification," *EURASIP J. Image Video Process.*, vol. 2017, no. 1, pp. 1–24, Dec. 2017.

[76] L. Kabbai, M. Abdellaoui, and A. Douik, "Image classification by combining local and global features," *Vis. Comput.*, vol. 34, no. 5, pp. 679–693, May 2019.

Dichao Liu received the B.S. degree from Nanjing University, China, in 2015, and the M.S. and Ph.D. degrees from Nagoya University, Japan, in 2018 and 2022, respectively. He is currently a Researcher with Navier, Inc., Japan. His current research interests include fine-grained image classification, fine-grained human action recognition, and image super-resolution.

Toshihiko Yamasaki (Member, IEEE) received the Ph.D. degree from The University of Tokyo, in 2004. He was a JSPS Fellow for Research Abroad and a Visiting Scientist at Cornell University from February 2011 to February 2013. He is currently a Professor with the Department of Information and Communication Engineering, Graduate School of Information Science and Technology, The University of Tokyo. His current research interests include attractiveness computing based on multimedia big data analysis, computer vision, pattern recognition, and machine learning.

Yu Wang (Member, IEEE) received the M.S. degree in information science and the Ph.D. degree in engineering from Nagoya University, in 2010 and 2013, respectively. He is currently an Associate Professor with the Center for Information and Communication Technology, Hitotsubashi University.

Kenji Mase received the B.E. degree in electrical engineering and the M.E. and Ph.D. degrees in information engineering from Nagoya University, in 1979, 1981 and 1992, respectively. He joined the Nippon Telegraph and Telephone Corporation NTT in 1981 and at the NTT Human Interface Laboratories. He was a Visiting Researcher at the Media Laboratory, MIT from 1988 to 1989. He has been with the Advanced Telecommunications Research Institute (ATR), from 1995 to 2002. He became a Professor at Nagoya University in August 2002. He is currently with the Graduate School of Informatics, Nagoya University. He has been a Research Supervisor of JST CREST on Symbiotic Interactions, since 2017. His research interests include gesture recognition, computer graphics, artificial intelligence and their applications for computer-aided communications, wearable/ubiquitous computers, and lifelog. He is a fellow of Institutes of Electronics, Information and Communication Engineers (IEICE) of Japan, and a member of the Information Processing Society of Japan (IPSJ), Japan Society of Artificial Intelligence (JSAI), Virtual Reality Society of Japan, Human Interface Society of Japan and ACM, and a Senior Member of IEEE Computer Society. He was a Section Chair of IEEE Nagoya Section from 2014 to 2015. He is the 24th and 25th Associate Member of Science Council of Japan.

Jien Kato (Senior Member, IEEE) received the M.E. and Ph.D. degrees in information engineering from Nagoya University, in 1990 and 1993, respectively. She became an Assistant Professor at Toyama University. She was a Visiting Researcher at the University of Oxford in 1999 for one year. She became an Associate Professor at the Graduate School of Engineering of Nagoya University in 2000. She has been a Professor with the College of Information Science and Engineering, Ritsumeikan University, since 2018. Her research interests include object recognition, visual event recognition, and machine learning. She is a member of IEICE, IPSJ and JSAI.