Computers and Electrical Engineering 110 (2023) 108861

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

A lightweight temporal attention-based convolution neural network for driver's activity recognition in edge☆

Lichao Yang, Weixiang Du, Yifan Zhao *

School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield, UK

ARTICLE INFO

This paper is for CAEE special section VSI-smtr. Reviews processed and recommended for publication to the Editor-in-Chief by Guest Editor Dr Muhammad Sulaiman

Keywords:
Ndra recognition
Efficient CNN
Attention mechanisms
Edge computing

ABSTRACT

Low inference latency and accurate response to environment changes play a crucial role in the automated driving system, especially in the current Level 3 automated driving. Achieving the rapid and reliable recognition of driver's non-driving related activities (NDRAs) is important for designing an intelligent takeover strategy that ensures a safe and quick control transition. This paper proposes a novel lightweight temporal attention-based convolutional neural network (LTA-CNN) module dedicated to edge computing platforms, specifically for NDRAs recognition. This module effectively learns spatial and temporal representations at a relatively low computational cost. Its superiority has been demonstrated in an NDRA recognition dataset, achieving 81.01% classification accuracy and an 8.37% increase compared to the best result of the efficient network (MobileNet V3) found in the literature. The inference latency has been evaluated to demonstrate its effectiveness in real applications. The latest NVIDIA Jetson AGX Orin could complete one inference in only 63 ms.

1. Introduction

Achieving driving safety is the ultimate goal of the automated driving system. However, in the current commercial automated driving level (level 3 defined by the Society of Automotive Engineers [1]), engagement in non-driving-related activities (NDRAs) during automated driving could reduce the driver's surrounding monitoring and situation awareness [2,3], which could bring a risk for the driver to take over the vehicle when they are requested, and cause accidents. In the reported accidents involving automated vehicles, the lack of situational awareness is the main factor where the driver does not have enough time to sense the environment and conducts proper manoeuvres to avoid the accident [4]. Therefore, monitoring the drivers' state and the activities they are engaging in is crucial for designing an intelligent human-machine interface (HMI) to improve their situation awareness before the takeover process.

The real driving scenarios are generally complex and uncertain, which requires the automated driving system to sense the environment, make decisions and perform the manoeuvre in a short time. The automated driving system integrates many subsystems for different tasks such as perception, localisation, etc. It has to collect and process data that are generally in a significant volume and highly heterogeneous from different types of sensors. Therefore, optimising the algorithm complexity or developing lightweight algorithms to achieve a lower decision-making latency is highly demanded to design the next generation of automated driving systems.

☆ This paper was recommended for publication by Associate Editor Dr Muhammad Sulaiman

* Corresponding author.
E-mail address: yifan.zhao@cranfield.ac.uk (Y. Zhao).

https://doi.org/10.1016/j.compeleceng.2023.108861
Received 18 March 2023; Received in revised form 2 July 2023; Accepted 3 July 2023
Available online 6 July 2023

Monitoring the driver's non-driving-related behaviours, as one of the subsystems, has been widely researched and developed in these years, which benefits from the rapid and significant progress of human action recognition made by the computer vision community. Yang et al. [5] mapped the driver's gaze into a view of the vehicle cabin and then combined it with object recognition to determine the visual-related NDRAs. Such a method has high prediction confidence of the NDRAs recognition since it directly locates the driver's visual attention. Nevertheless, it lacks the capability of classifying the activities with the same object. Xing et al. [6] extracted the driver's head rotation angles and the joint positions of the upper body, and then used a feedforward neural network to classify NDRAs. Similarly, Martin et al. [7] used the joint positions of the upper body and the image of the movement as the inputs and then employed three recurrent neural networks (RNNs) to detect NDRAs. Apart from the skeleton features, Xing et al. [8] further extracted the driver's upper body through image segmentation and used a convolution neural network (CNN) to recognise NDRAs. All of these methods adopted the hand-crafted feature extraction and followed by the neural network-based classifier. Yang et al. [9] employed the CNN-based ResNet-50 to extract the features from images and combined the optical flow of the images, which presents the driver's hand movement to achieve the recognition of NDRAs. Eraqi et al. [10] captured the image inside the vehicle cabin. They employed multiple CNNs with the inputs of the raw image, hand image, face image and skin-segmented image. Then they used a genetic algorithm to achieve the weighted ensemble classification. Such methods are usually based on the image input, which is mainly in the spatial domain and lacks the temporal representation of the driver's behaviour during the NDRAs engagement. Yang et al. [11] employed a dual-stream 3D CNN, which extracts the spatiotemporal representation of the driver's behaviour with the designed short-time spatial block and small-region temporal block, to recognise the NDRAs in the video stream. Instead of using 3D convolution, another way to learn spatial-temporal features from videos is spatiotemporal attention mechanisms. Such a method usually employs CNN for spatial representation learning since it shows excellent capability in the spatial domain. In the temporal domain, the temporal features are encoded by sequence-based signal processing methods like Recurrent Neural Networks and Long Short-Term Memory [12, 13]. However, the 3D convolution is normally computationally costly, and the structure of the Spatiotemporal attention mechanisms-based method is complex. Existing studies focus more on inference accuracy than algorithms' computation complexity. There is a lack of efficient network architecture that monitors the driver's behaviour and recognises NDRAs from videos. Furthermore, the existing studies are mainly based on offline analysis. The methods were developed and tested on high-performance workstations with GPU, which cannot be directly used in real road-testing scenarios. Cloud computing to implement these algorithms is a challenge for such an application due to the high volume of data, the requirement for a rapid response and data security risk [14]. The evaluation of the computation complexity of the existing methods is mainly on the calculation operation demands. There is a lack of evaluation of the inference latency, which refers to the time cost for inferring one instance, on the on-vehicle edge computing devices. This type of evaluation is essential because the computation power of the deployment platform, algorithm complexity and data transmission could all affect the inference latency in real applications.

This paper proposed a novel lightweight 3D CNN-based temporal attention module, called LTA-CNN, to achieve low latency inference on the on-vehicle edge device in video-based NDRA recognition. Unlike the conventional 3D convolution module with a limited receptive field, the proposed module models the global information in the time domain. Specifically, the proposed module uses the 3D convolution operation in the spatial domain and further employs the attention mechanisms-based temporal weighting function to enhance the representation in the temporal domain. This module tends to achieve high accuracy with much less computational cost. The proposed module can be trained end-to-end and used as a plugin module for the existing efficient 3D CNN. Moreover, the saliency map is employed to visualise the features learned in the hidden layer of the network to validate its capability of learning the semantic representations of the activities. To further evaluate the applicability of the model in real driving scenarios, the performance regarding inference latency and accuracy of the proposed LTA-CNN and several state-of-the-art methods has been compared on four types of edge computing devices from the family of the NVIDIA Jetson AI platform, which have been widely used in the fields such as robotics and automated driving, etc. [15–17].

In summary, the contribution of this study includes:

- a novel lightweight temporal attention module for video analysis to extract spatial-temporal features effectively with low computational cost
- the application of the proposed LTA-CNN on the NDRAs classification
- the comparison of the proposed network and other SOTA methods on four edge computing devices to evaluate the inference cost for real in-vehicle applications
- visualisation of the extracted features through extending Grad-CAM++ [18] to video data.

The rest of this paper is organised as follows: Section 2 introduces the design of the proposed network, employed dataset and hardware. Section 3 details the state-of-the-art models that were used for comparison and the training process. Section 4 presents the model performance results and the visualisation of features that the model extracted. Finally, the conclusion is given in Section 5.

2. Methodology

This section is divided into three parts. Section 2.1 elaborates on the proposed LTA-CNN network used to classify the NDRAs and the method for visualising the learned spatial-temporal representation. Section 2.2 shows the NDRAs dataset used to evaluate the proposed method. Section 2.3 further introduces edge computing devices employed for testing the on-vehicle inference latency in real-world applications.

2.1. LTA-CNN and feature visualisation

The details of the network will be illustrated in the following sections. Section 2.1.1 introduced the proposed novel lightweight temporal attention-based module. Section 2.1.2 finally shows the whole structure of the proposed LTA-CNN network. Section 2.1.3 illustrates the technique used to visualise the learned features in the network.

2.1.1. lightweight temporal attention-based module

The proposed novel lightweight temporal attention-based module is presented in Fig. 1. It introduces the attention mechanism, which compresses the global information and extracts the critical features in both channel and temporal levels in an efficient manner. The proposed module aims to significantly reduce the computational cost meanwhile preserving model performance in accuracy, compared to the conventional 3D CNN. Specifically, the module used a $1 \times 1 \times 1$ convolution kernel to expand the channel dimension. A $1 \times 3 \times 3$ depthwise convolution kernel is then employed to extract the spatial features, and followed by a channel weighting function is employed to improve the learned representation at the channel level. Then a temporal weighting function is introduced to extract the most valuable representation in the time domain among the weighted channels. In the end, a $1 \times 1 \times 1$ convolution kernel is used to compress the dimension. From now on, the 3D convolution kernel is denoted as $d \times k \times k$, where d and k are the temporal depth and the spatial size, respectively.

Inverted Linear Bottleneck block [19] is designed for efficient neural network-based models, which employs the depthwise separable convolution to reduce the computational cost of the conventional convolution and further uses more channels to improve the feature extraction capability. It employed the linear activation function at the end of the bottleneck block to reduce the inevitable information loss of spatial information in low-dimensional space encoding [19].

Depthwise separable convolution factorises the conventional convolution into two operations, depthwise convolution and pointwise convolution. The depthwise convolution only applies the convolution operation for each single input channel. After that, the pointwise convolution, $1 \times 1 \times 1$ convolution, will build a linear combination of the output of the depthwise convolution.

The n^{th} feature map O of the conventional convolution can be calculated as:

$$O_{(i,j,h,n)} = \sum_{k,l,m,c}^{K,L,M,C} W_{(k,l,m,n)} \cdot F_{(k+i,l+j,m+h,c)} \tag{1}$$

The n^{th}feature map O_{dw} of the depthwise convolution for a single input channel can be calculated as:

$$O_{dw(i,j,h,n)} = \sum_{k,l,m}^{K,L,M} W_{(k,l,m,n)} \cdot F_{(k+i,l+j,m+h,n)} \tag{2}$$

The feature map O_{pw} of the following pointwise convolution can be calculated as:

$$O_{pw(i,j,h,n)} = \sum_{c}^{C} W_{(n)} \cdot F_{(i,j,h,c)} \tag{3}$$

where i, j, h and k, l, m are the 3-dimensional location indicators. K, L are the spatial size of the kernel, M is the temporal size of the kernel. W is the convolutional kernel, F is the input feature map, C is the number of the input channel.

The computation ratio R_{ds} between depthwise separable convolution and conventional convolution for 3D CNN can be expressed as:

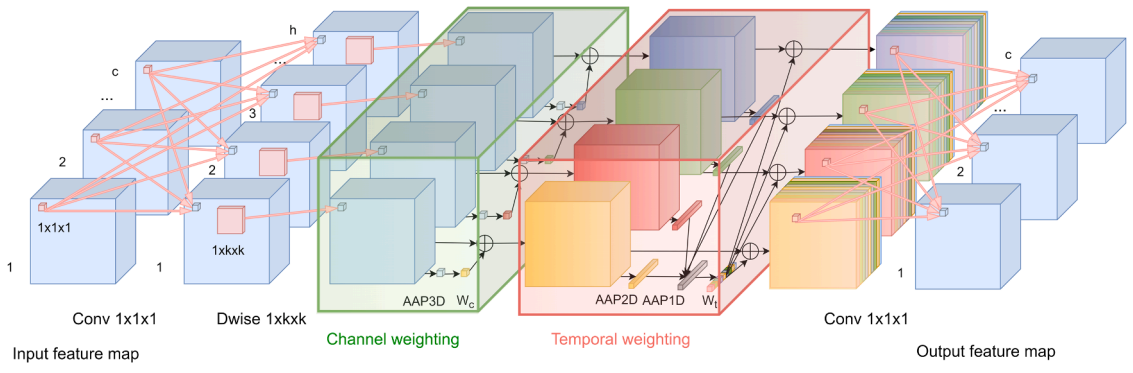

Fig. 1. Proposed lightweight temporal attention-based module structure. where k is the kernel size and h is the number of the channels for depthwise convolution.

$$R_{ds} = \frac{S_W \cdot S_W \cdot S_W \cdot C \cdot S_{O_i} \cdot S_{O_j} \cdot S_{O_h} + N \cdot C \cdot S_{O_i} \cdot S_{O_j} \cdot S_{O_h}}{S_W \cdot S_W \cdot S_W \cdot N \cdot C \cdot S_{O_i} \cdot S_{O_j} \cdot S_{O_h}} = \frac{1}{N} + \frac{1}{S_W^3} \tag{4}$$

where S_W is the kernel size and $S_{Oi} \cdot S_{Oj} \cdot S_{Oh}$ are the size of the output feature map in 3-dimension.

To reduce the computational complexity of the 3D convolutional operation, we factorise the 3D convolution operation into 2D spatial convolution and 1D temporal convolution in this paper. Instead of using a $3 \times 3 \times 3$ depthwise convolution kernel, a $1 \times 3 \times 3$ depthwise convolution kernel and a 1D temporal weight function have been employed to extract the spatiotemporal features. Inspired by the Squeeze-and-Excitation [20], channel weighting has been employed to compute the channel-wise importance.

The channel weighting function is a way to improve channel interdependencies with very limited computational cost. It provides the attention mechanism at the channel level. Such a module compresses the spatial and temporal information in a single channel and then adds a channel weight functionC_W, to achieve the fusion of the spatiotemporal and the channel information. The weighting function C_W can be expressed as:

$$C_W(F_{se},\ W_c) = \sigma(W_{c_2}\delta(W_{c_1}AAP_{3D}(F_{se}))) \tag{5}$$

where $W_{c_1} \in \mathbb{R}^{\frac{c}{4} \times c}$ and $W_{c_2} \in \mathbb{R}^{c \times \frac{c}{4}}$, F_{se} is the input feature map for the module.

The temporal weight function T is designed based on the attention mechanism, which compresses the features in the spatial domain and weights the temporal information in a global way to extract the valuable temporal representation rather than focusing on the limited receptive field in the time domain. Such a design could further reduce the computational complexity and improve the capability of feature extraction in the time domain. The output of the temporal weight function $O_t = (O_{t_1}, O_{t_2}, \ldots, O_{t_c})$ can be expressed as:

$$(O_{t_1}, O_{t_2}, \ldots, O_{t_c}) = T(F_{t_1}, F_{t_2}, \ldots, F_{t_c}) \tag{6}$$

where c is the number of the feature map channel. The input of the weight function is applied with a 2D adaptive average pooling (AAP_{2D}) first to compress the spatial information. Then a 1D adaptive average pooling (AAP_{1D}) is employed, which is applied at the channel level to integrate the channel information. The pooled output goes through two fully connected layers. ReLU and sigmoid activation function was employed to add nonlinearity for the first and second fully connected layer, respectively. The weight function T can be calculated as:

$$T(F_t,\ W_t) = \sigma(W_{t_2}\delta(W_{t_1}AAP_{1D}(AAP_{2D}(F_t)))) \tag{7}$$

where δ and σ are the ReLU and sigmoid functions. $W_{t_1}, W_{t_2} \in \mathbb{R}^{t \times t}$.

The computation ratio R_{tw} between the temporal weight-enabled depthwise separable convolution and the 3D depthwise convolution can be expressed as:

$$R_{tw} = \frac{S_W \cdot S_W \cdot C \cdot S_{O_i} \cdot S_{O_j} \cdot S_{O_h} + C \cdot S_{O_h} \cdot S_{O_h} \cdot S_{O_h}}{S_W \cdot S_W \cdot S_W \cdot C \cdot S_{O_i} \cdot S_{O_j} \cdot S_{O_h}} \tag{8}$$

2.1.2. Model structure

The specification of the model is presented in Fig. 2. The inputs, including a stack of head movement frames and a stack of hand movement frames, are fed into convolution layer1 for the initial feature extraction separately. Then we concatenate the outputs together and go through 6 convolution layers with the designed block. The number of the blocks and input channels, kernel size, stride and the number of the expanded hidden channels in the block are presented in Fig. 2. A pooling layer is employed after the feature extraction to adjust the dimension of the output feature. Two fully connected layers are used for the final classification.

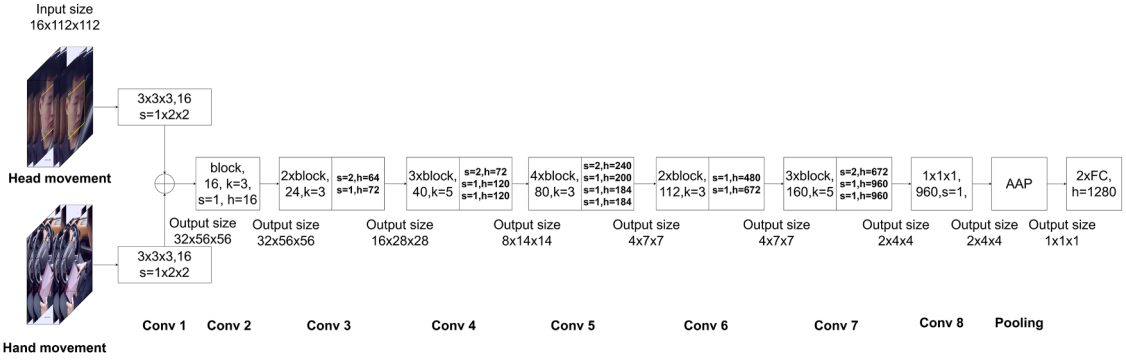

Fig. 2. Model specification of the proposed LTA-CNN. In this figure, s is the stride of the convolution operation, h is the number of the channel for the depthwise convolution layer. AAP refers an adaptive average pooling layer and FC stands for the fully connected layer.

2.1.3. Saliency map visualisation

The visualisation of the feature map inside the CNN is considered as a way to understand the black-box model, which has been widely researched in recent years [18,21,22]. The visualised saliency map highlights the region of the input that is important for the prediction, which provides a visual explanation of the learned features. In this study, Grad-CAM++ [18] was used to produce the saliency map, which computes the weighting of the gradients of each pixel in the feature map for the final classification to calculate the importance at the pixel level.

The final prediction score Y^c for the class c can be expressed as:

$$Y^c = \sum_k w_k^c \sum_i \sum_j \sum_h A_{ijh}^k \tag{9}$$

where (i, j, h) is a specific location in the feature map A_{ijh}^k, k is the number of the feature map A^k, and w_k^c is its weight for the class c.

The weight w_k^c is calculated by taking a weighted average of the pixel-wise gradients, which can be written as:

$$w_k^c = \sum_i \sum_j \sum_h \alpha_{ijh}^{kc} relu\left(\frac{\partial Y^c}{\partial A_{ijh}^k}\right) \tag{10}$$

where α_{ijh}^{kc} is the weighting coefficients and $\frac{\partial Y^c}{\partial A_{ijh}^k}$ is the pixel-wise gradient for the feature map A^k and class c.

Eq. (9) can be rewritten based on Eq. (10):

$$Y^c = \sum_k \left[\sum_a \sum_b \sum_d \alpha_{abd}^{kc} relu\left(\frac{\partial Y^c}{\partial A_{abd}^k}\right)\right] \sum_i \sum_j \sum_h A_{ijh}^k \tag{11}$$

where (a, b, d) and (i, j, h) are the separate iterators for the same feature map A^k, which are used for avoiding confusion in the summation process. *relu* was dropped in the derivation since it is only a threshold for the gradients flowing back. Taking partial derivative A_{ijh}^k on both sides:

$$\frac{\partial Y^c}{\partial A_{ijh}^k} = \sum_a \sum_b \sum_d \alpha_{abd}^{kc} \frac{\partial Y^c}{\partial A_{abd}^k} + \sum_a \sum_b \sum_d A_{abd}^k \left(\alpha_{ijh}^{kc} \frac{\partial^2 Y^c}{\left(\partial A_{ijh}^k\right)^2}\right) \tag{12}$$

Then taking a further partial derivative A_{ijh}^k on both sides, since (a, b, d) and (i, j, h) are the iterators for the same feature map, Eq. (12) can be written as:

$$\frac{\partial^2 Y^c}{\left(\partial A_{ijh}^k\right)^2} = 2\alpha_{ijh}^{kc} \frac{\partial^2 Y^c}{\left(\partial A_{ijh}^k\right)^2} + \sum_a \sum_b \sum_d A_{abd}^k \left(\alpha_{ijh}^{kc} \frac{\partial^3 Y^c}{\left(\partial A_{ijh}^k\right)^3}\right) \tag{13}$$

Rearranging terms, α_{ijh}^{kc} can be rewritten as:

$$\alpha_{ijh}^{kc} = \frac{\frac{\partial^2 Y^c}{\left(\partial A_{ijh}^k\right)^2}}{2\frac{\partial^2 Y^c}{\left(\partial A_{ijh}^k\right)^2} + \sum_a \sum_b \sum_d \left(A_{abd}^k \frac{\partial^3 Y^c}{\left(\partial A_{ijh}^k\right)^3}\right)} \tag{14}$$

The w_k^c in Eq. (10) can be calculated based on Eq. (13). Then the class-based saliency map M^c can be calculated as:

$$M^c = relu\left(\sum_k w_k^c A^k\right) \tag{15}$$

2.2. Dataset and pre-processing

The NDRA recognition dataset used in this study contains four types of NDRAs and two types of driving activities (DAs). Based on a tablet, four NDRAs include Answering questionnaires, Playing games, Reading news and Watching movies. Two DAs are Checking road and Driving. This dataset has 14 subjects (12 male and two female) whose ages range from 23 to 35. The subjects are from 8 counties. The data also cover different weather and illumination conditions, including sunny, cloudy, rainy and snowy.

The data for each activity includes two feeds, the driver's head movement and hand movement. The pair of both feeds are considered as one instance. Each feed contains 16 adjacent frames extracted from a video captured at 30 fps. There are, in total, 7960 instances for six activities in this dataset, which have been randomly divided into five splits based on the subjects for the cross-validation. In each split, the data of 11 subjects were used for training and the rest of 3.

2.3. Hardware

NVIDIA Jetson is a high-performance AI platform for edge computing. Four Jetson modules (Fig. 3), including Jetson Nano, Jetson TX2 and Jetson AGX Xavier, were used to test the inference latency of the model for further in-vehicle implementation. Jetson Nano is an entry-level AI development module that can process multiple neural networks in parallel with the data acquired from high-resolution sensors. Jetson Nano is a small development board with limited computation power. Jetson TX2 upgrades the power efficiency and performance to another level than Nano. Jetson AGX Xavier is an edge computer designed to deploy end-to-end AI robotics applications. Jetson AGX Orin is the latest and most powerful AI computer for energy-efficient autonomous machines. It provides the hardware acceleration for the entire AI pipeline and multiple high-speed inputs and outputs. The comparison of the technical specification is shown in Table 1.

3. Comparison and training process

Several classical efficient CNNs and state-of-the-art backbones were revised to 3D convolutions and compared with the proposed network, including

- MobileNet V1 [23]: it replaces the conventional convolution with depthwise separable convolution in the network to reduce the computational cost.
- MobileNet V2 [19]: it utilises the inverted residual structure to enhance the capability of feature learning and removes the non-linear activation in the narrow layers to maintain representational power.
- ShuffleNet V1 [24]: it employs the pointwise group convolution and channel shuffle operation to address the computational expensiveness of the pointwise convolutions.
- ShuffleNet V2 [25]: it further introduces a channel split operator to decrease the latency on the work device.
- MobileNet V3 [26]: it is produced by the network architecture search techniques, which strike the best trade-off between performance and latency. It also employs the squeeze and excitation structure to improve accuracy.

The methods' performance was evaluated with multiple model sizes controlled by the channel multiplier, which is used to adjust the channel number in the convolutional layer. The channel multiplier was set as 0.5, 1, 1.5, and 2. Since the input of the network is a pair of head and hand movements, the networks extract the features separately and combine both feeds at a high level with the adaptive average pooling before the classifier. The model size and the computational cost for all the networks are presented in Table 2. It can be seen that the proposed LTA-CNN has a similar level of model size as MobileNet V2.

In the training process, Adam was adapted as a parameter optimisation with a mini-batch size of 64. The initial learning rate was set as 0.001, and the learning rate adaptive decay method has been employed to adjust the learning rate adaptively to ensure an efficient optimisation process. The whole training epoch was set as 60.

4. Result

4.1. Model performance

The performance of the evaluated models has been presented in Fig. 4. With the channel multiplier increase from 0.5 to 2, the classification accuracy of the models also increases from 22% (ShuffleNet V1) to 7% (MobileNet V3). The MobileNet V2, V3 and

Fig. 3. Edge computing modules used in the latency test. Top eft: Jetson Nano. Top Right: Jetson AGX Xavier. Bottom left: Jetson TX2. Bottom right: Jetson AGX Orin.

Table 1
Technical specifications of the tested edge computing devices.

	Nano	TX2	AGX Xavier	AGX Orin
CPU	Quad-Core Arm® Cortex®-A57 MPCore processor	Dual-Core NVIDIA Denver 2 64-Bit CPU and Quad-Core Arm® Cortex®-A57 MPCore processor	8-core NVIDIA Carmel Arm®v8.2 64-bit CPU	12-core Arm® Cortex®-A78AE v8.2 64-bit CPU
GPU	128-core NVIDIA Maxwell™ GPU	256-core NVIDIA Pascal™ GPU	512-core NVIDIA Volta™ GPU with 64 Tensor Cores	2048-core NVIDIA Ampere GPU with 64 Tensor Cores
Memory (GB)	4	8	32	64
Power (W)	5 | 10	7.5 | 15	10 | 15 | 30	15 | 30 | 50| 60

Table 2
Comparison of the model size and the computational cost with different model size.

Model	Parameters ($\times 10^6$)				Floating point operations per second (FLOPs) ($\times 10^9$)			
Channel multiplier	0.5	1	1.5	2	0.5	1	1.5	2
MobileNet V1	1.73	6.6	14.61	25.76	0.19	0.48	0.85	1.32
MobileNet V2	1.30	4.3	9.33	16.28	0.42	1.12	2.1	3.37
ShuffleNet V1	0.52	1.89	4.13	7.22	0.15	0.4	0.69	1.07
ShuffleNet V2	0.53	2.13	4.37	8.83	0.25	0.39	0.58	0.87
MobileNet V3	2.51	7.68	16.71	28.91	0.31	0.73	1.45	2.22
LTA-CNN	1.45	4.31	9.64	16.94	0.25	0.63	1.29	2.02

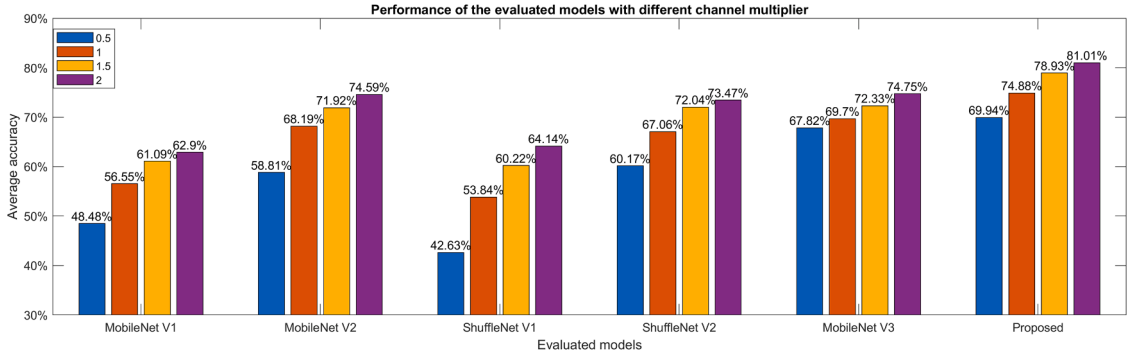

Fig. 4. Average accuracy of all the splits for evaluated models with a set of channel multiplier (0.5, 1, 1.5, 2).

ShuffleNet V2 have a similar level of performance when the channel multiplier is larger than 0.5, which is around 68% (1), 72% (1.5) and 74% (2). However, when the channel multiplier is 0.5, MobileNet V3 shows better performance (67.82%) than the other two models. ShuffleNet V1 and MobileNet V1 also perform similarly when the model is relatively large. ShuffleNet V1 (0.5) shows the lowest accuracy (42.63%), which is due to the limited number of channels that restricts its capability of feature extraction. Compared to other models, the proposed network shows the best performance among all the channel multipliers (model size and complexity) and achieves 81.01% average accuracy when the multiplier is 2. Compared to the best result for the SOTA the performance improvements in terms of different multipliers are 3.13% (multiplier: 0.5), 7.43% (multiplier: 1), 9.12% (multiplier: 1.5), and 8.37% (multiplier: 2). To present the generalisation capability of the model, Table 3 shows the results of the cross-validation for each split when the multiplier is 2. It can be observed that the standard deviation of all the models is under 1.5%. The values for ShuffleNets are below 1%. The standard deviation for the rest modes is around 1.1%.

Table 3
Performance of the model when the channel multiplier set as 2 for all splits in dataset.

	Accuracy					
	Split1	Split2	Split3	Split4	Split5	Mean & Std
MobileNet V1	62.14%	61.68%	62.77%	64.21%	63.69%	62.90% ± 1.05%
MobileNet V2	73.99%	74.63%	75.87%	72.84%	75.64%	74.59% ± 1.24%
ShuffleNet V1	63.32%	64.25%	64.09%	65.69%	63.34%	64.14% ± 0.97%
ShuffleNet V2	73.90%	74.41%	73.24%	72.25%	73.54%	73.47% ± 0.81%
MobileNet V3	74.52%	75.32%	73.56%	74.05%	76.38%	74.75% ± 1.10%
LTA-CNN	80.32%	79.80%	81.70%	82.56%	80.665	81.01% ± 1.11%

To evaluate the latency, the models were evaluated on four edge computing devices, and the results are shown in Table 4. Due to the limited computational capability of the GPU on Jetson Nano, only the smallest model (channel multiplier is 0.5) can be implemented. MobileNet V1 can not be implemented because of its poor memory efficiency on GPU. The proposed method achieves a relatively low latency, which is 417 ms, among all the evaluated models. For the Jetson TX2, ShuffleNet V1 achieves the least latency across all sizes of the model. The proposed method has a slightly higher latency than ShuffleNet V1 and is below 0.5 s for the largest model. MobileNet V2 achieves the highest latency. Similar results can be found in the Jetson AGX Xavier implementation. However, with the increase of the computation capability of GPU on the device, ShuffleNet V1 shows a slightly faster inference than the proposed method. On this device, the proposed method could complete at least five inferences in a second. Combining Tables 2 and 4, it can be observed that the computational complexity (FLOPS) of the network dominates the cost of the inference. However, when deployed on GPUs, the proper design of the network could speed up the inference process and further reduce the latency. For instance, the proposed LTA-CNN has a higher computational complexity (FLOPS) than MobileNet V1 across all sizes of the model, but it costs less inference time on Jetson TX2. Figs. 5 and 6 plot the model's accuracy against latency on Jetson TX2 and Jetson AGX Xavier, respectively. It can be seen that as the model becomes larger, the inference latency and accuracy also increase. On a different device, the overall performance could be different. For instance, on Jetson TX2, MobileNet V3 achieves a higher accuracy with a similar level of latency than ShuffleNet V2. However, on Jetson AGX Xavier, the inference latency of ShuffleNet V2 is much less than MobileNet V3, which is crucial for NDRA recognition. In both figures, the proposed model outperforms all the evaluated state-of-the-art models. Benefits from the strong computation power of the Jetson AGX Orin, the inference latency can be reduced to around 60 ms. The proposed method shows a similar level of inference latency compared with other state-of-the-art, but the superior inference accuracy is obvious. Fig. 7 presents a performance comparison between the evaluated efficient models with the conventional 3D CNN models in terms of model size. It has been observed that DS3D [11] achieves the best classification accuracy. In terms of accuracy, the proposed LTA-CNN is only 2.5% less than DS3D. However, it only has a 75% model size of DS3D. The FLOPs of the proposed LTA-CNN is only 2.8% of DS3D.

4.2. Saliency may visualisation

This section visualises the features learned in the hidden layer through the saliency map. The input of the network contains both head and hand movements. Due to the data protection policy, the data containing facial information is not included in this section. The 16 hand movement frames are downsampled to 8 for an easy display. The class-discriminative saliency map of the last convolution layer in Conv4 is presented in Fig. 8. The results of ShuffleNet V2 (SV2) and MobileNet V3 (MV3) are used for comparison. It can be seen that, for the first 3 NDRAs, the hand movement has been highlighted in SV2. However, it contains more noise compared with the other two methods. For answering questionnaires and playing games, both MV3 and the proposed method learned the spatiotemporal features of hand movement. The results of the proposed method show a higher sensitivity in the time domain. It highlights a couple of frames with the key finger movement in the action classification, while the features learned by MV3 cover a longer period. For reading, the MV3 model mainly focuses on the tablet location, while the proposed method covers both tablet and hand features. For the activity of watching videos, the learned features of SV2 are mainly from a part of the tablet, and the MV3 highlights the region which is above the driver's hand. The features that the proposed method learned are the spatial relationship between both hands and the tablet. For driving, SV2 captures the driver's both hands movements. Even though MV3 highlights the driver left arm movement, the main focus is around the door. The proposed method focuses on the driver's right-hand movement. In summary, comparing these two models, the proposed method shows a higher semantic relevance of the extracted spatiotemporal features. The proposed temporal attention module presents a stronger capability of extracting semantic representation of the hand movement in the time domain.

5. Conclusions

In this paper, an efficient and low-latency CNN-based temporal attention module has been proposed, which learns the spatio-temporal representation through spatial convolution and attention-enhanced temporal weighting. Unlike the conventional 3D convolution operation, the proposed module could enhance the learned representation in the temporal domain with lower computational complexity. In this study, the performance of the proposed LTA-CNN has been evaluated on an NDRAs recognition dataset. The results demonstrate a significant improvement in accuracy against several state-of-the-art methods. The saliency map of the learned features shows its capability of extracting the key spatiotemporal representation from the activity, specifically in the time domain. Moreover, the proposed module is end-to-end trainable and can be used as a plugin module for the existing 3D CNN network, which has

Table 4
Comparison of latency for different device and different channel multiplier.

	Latency (ms)												
Device	Nano	TX2				AGX Xavier				AGX Xavier Orin			
Channel Multiplier	0.5	0.5	1	1.5	2	0.5	1	1.5	2	0.5	1	1.5	2
MobileNet V1	N/A	304	397	482	531	98	131	148	163	42	42	41	42
MobileNet V2	634	339	488	679	867	133	206	232	278	56	56	56	61
ShuffleNet V1	532	251	275	300	361	114	119	132	145	58	58	60	61
ShuffleNet V2	374	243	355	541	772	116	128	145	162	56	60	63	64
MobileNet V3	509	323	423	507	686	148	168	210	258	69	70	69	71
LTA-CNN	417	262	297	376	458	121	135	155	174	62	62	63	63

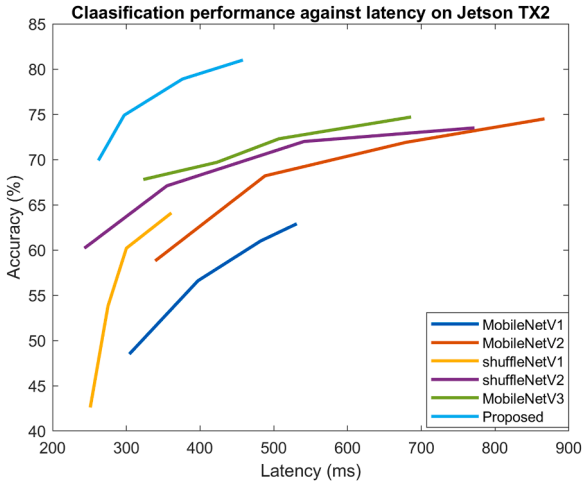

Fig. 5. Classification and inference performance of the model on Jetson TX2.

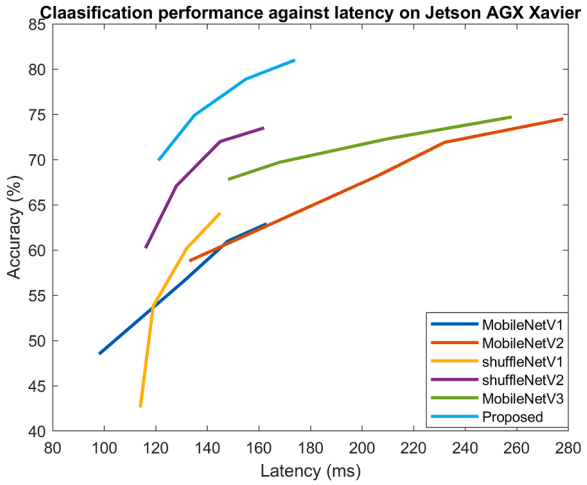

Fig. 6. Classification and inference performance of the model on Jetson AGX Xavier.

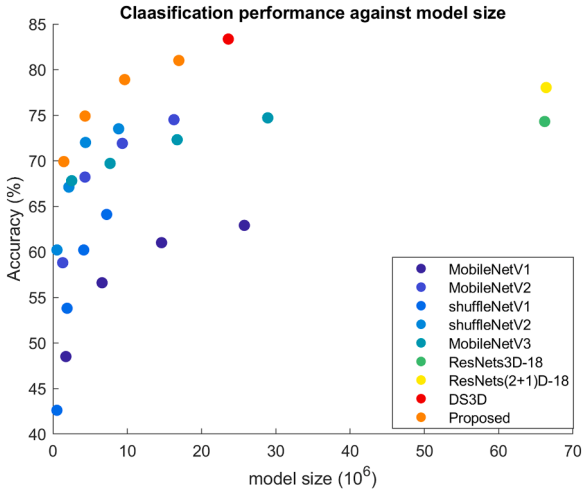

Fig. 7. Classification performance against the model size.

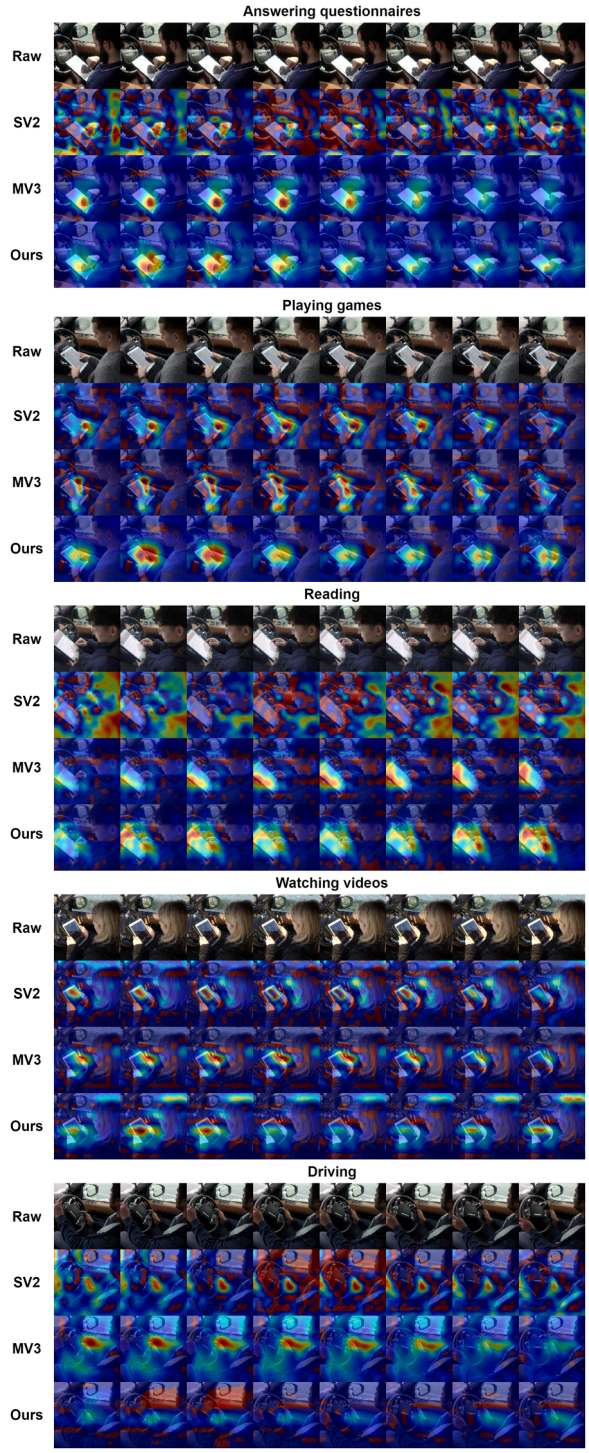

Fig. 8. Class-discriminative saliency maps of the last convolutional layer of Conv4 produced by Grad-CAM++ [18] for all NDAs. The first row of each activity is the raw frames imported into the network. The red regions refer to a higher association with the final classification while the regions in blue show the weak relevance.

the applicability for lightweight video-based general action recognition. Furthermore, this paper tested the latency performance of the algorithms on four widely-used edge computing devices for the evaluation of the algorithm's applicability in real driving scenarios. The results demonstrated that the inference latency of the algorithm does not only depend on the computation complexity but also on the hardware computational power, data transition speed, etc. Therefore, it is important to evaluate the model latency for the real

application. In the Jetson TX2 and AGX Xavier, the proposed method presents the competitive inference latency due to the design of the network compared to the state-of-the-art. Although the advance of the latency for the proposed network in the most powerful device, Jetson AGX Orin, is not significant, in such a device, the latency can be reduced to 63 ms. The proposed method could achieve fast and reliable NDRAs recognition.

Funding

This work was supported by the Royal Academy of Engineering Industrial Fellowship [#grant IF2223B-110].

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.compeleceng.2023.108861.

References

[1] SAE, J3016. Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems. 2021.
[2] de Winter JCF, Happee R, Martens MH, Stanton NA. Effects of adaptive cruise control and highly automated driving on workload and situation awareness: a review of the empirical evidence. Transp Res Part F Traffic Psychol Behav 2014;27:196–217. https://doi.org/10.1016/j.trf.2014.06.016.
[3] Yoon SH, Ji YG. Non-driving-related tasks, workload, and takeover performance in highly automated driving contexts. Transp Res F Traffic Psychol Behav 2019;60:620–31. https://doi.org/10.1016/j.trf.2018.11.015.
[4] Song Y, Chitturi MV, Noyce DA. Automated vehicle crash sequences: patterns and potential uses in safety testing. Accid Anal Prev 2021;153:106017. https://doi.org/10.1016/j.aap.2021.106017.
[5] Yang L, Dong K, Ding Y, Brighton J, Zhan Z, Zhao Y. Recognition of visual-related non-driving activities using a dual-camera monitoring system. Pattern Recognit 2021;116:107955. https://doi.org/10.1016/j.patcog.2021.107955.
[6] Xing Y, Lv C, Zhang Z, Wang H, Na X, Cao D, Velenis E, Wang F. Identification and analysis of driver postures for in-vehicle driving activities and secondary tasks recognition. IEEE Trans Comput Soc Syst 2018;5:95–108. https://doi.org/10.1109/TCSS.2017.2766884.
[7] Martin M, Popp J, Anneken M, Voit M, Stiefelhagen R. Body pose and context information for driver secondary task detection. In: Proceedings of the 2018 IEEE intelligent vehicles symposium. 5. IEEE; 2018. p. 2015–21. https://doi.org/10.1109/IVS.2018.8500523.
[8] Xing Y, Lv C, Wang H, Cao D, Velenis E, Wang FY. Driver activity recognition for intelligent vehicles: a deep learning approach. IEEE Trans Veh Technol 2019;68:5379–90. https://doi.org/10.1109/TVT.2019.2908425.
[9] Yang L, Yang T, Liu H, Shan X, Brighton J, Skrypchuk L, Mouzakitis A, Zhao Y. A refined non-driving activity classification using a two-stream convolutional neural network. IEEE Sens J 2020;21:1. https://doi.org/10.1109/JSEN.2020.3005810. –1.
[10] Eraqi HM, Abouelnaga Y, Saad MH, Moustafa MN. Driver distraction identification with an ensemble of convolutional neural networks. J Adv Transp 2019;2019:1–12. https://doi.org/10.1155/2019/4125865.
[11] Yang L, Shan X, Lv C, Brighton J, Zhao Y. Learning spatio-temporal representations with a dual-stream 3D residual network for non-driving activity recognition. IEEE Trans Ind Electron 2021;0046:1. https://doi.org/10.1109/TIE.2021.3099254. –1.
[12] Meng L, Zhao B, Chang B, Huang G, Sun W, Tung F, Sigal L. Interpretable spatio-temporal attention for video action recognition. In: Proceedings of the 2019 IEEE/CVF international conference on computer vision workshop; 2019. p. 1513–22. https://doi.org/10.1109/ICCVW.2019.00189. IEEE.
[13] Yan C, Tu Y, Wang X, Zhang Y, Hao X, Zhang Y, Dai Q. STAT: spatial-temporal attention mechanism for video captioning. IEEE Trans Multimed 2020;22:229–41. https://doi.org/10.1109/TMM.2019.2924576.
[14] Balasubramaniam S, Vijesh Joe C, Sivakumar TA, Prasanth A, Satheesh Kumar K, Kavitha V, Dhanaraj R. Optimization enabled deep learning-based DDoS attack detection in cloud computing. Int J Intell Syst 2023;2023:1–16. https://doi.org/10.1155/2023/2039217.
[15] Tang J., Liu S., Yu B., Shi W. PI-Edge: A low-power edge computing system for real-time autonomous driving services 2018:1–13.
[16] Liu S, Liu L, Tang J, Yu B, Wang Y, Shi W. Edge computing for autonomous driving: opportunities and challenges. Proc IEEE 2019;107:1697–716. https://doi.org/10.1109/JPROC.2019.2915983.
[17] Patrikar DR, Parate MR. Anomaly detection using edge computing in video surveillance system: review. Int J Multimed Inf Retr 2022;11:85–110. https://doi.org/10.1007/s13735-022-00227-8.
[18] Chattopadhay A, Sarkar A, Howlader P, Balasubramanian VN. Grad-CAM++: generalised gradient-based visual explanations for deep convolutional networks. In: Proceedings of the 2018 IEEE winter conference on applications of computer vision. 2018-Janua; 2018. p. 839–47. https://doi.org/10.1109/WACV.2018.00097. IEEE.
[19] Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC. MobileNetV2: inverted residuals and linear bottlenecks. In: Proceedings of the 2018 IEEE/CVF conference on computer vision and pattern recognition; 2018. p. 4510–20. https://doi.org/10.1109/CVPR.2018.00474.
[20] Hu J, Shen L, Albanie S, Sun G, Wu E. Squeeze-and-excitation networks. IEEE Trans Pattern Anal Mach Intell 2020;42:2011–23. https://doi.org/10.1109/TPAMI.2019.2913372.
[21] Springenberg J.T., Dosovitskiy A., Brox T., Riedmiller M. Striving for simplicity: the all convolutional net. 3rd int conf learn represent ICLR 2015 - work track proc 2014:1–14.

[22] Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: visual explanations from deep networks via gradient-based localization. Int J Comput Vis 2020;128:336–59. https://doi.org/10.1007/s11263-019-01228-7.
[23] Howard A.G., Zhu M., Chen B., Kalenichenko D., Wang W., Weyand T., Andreetto M., A.H. MobileNets: Efficient convolutional neural networks for mobile vision applications 2017.
[24] Zhang X, Zhou X, Lin M, ShuffleNet SJ. An extremely efficient convolutional neural network for mobile devices. In: Proceedings of the 2018 IEEE/CVF conference on computer vision and pattern recognition; 2018. p. 6848–56. https://doi.org/10.1109/CVPR.2018.00716.
[25] Ma N., Zhang X., Zheng H.T., Sun J. ShuffleNet V2: practical guidelines for efficient CNN architecture design. Lect notes comput sci (Including subser lect notes artif intell lect notes bioinformatics) 2018; 11218 LNCS:122–38. 10.1007/978-3-030-01264-9_8.
[26] Howard A, Sandler M, Chen B, Wang W, Chen LC, Tan M, Wang W, Zhu Y, Pang R, Vasudevan V, Le Q, Adam H. Searching for mobilenetV3. In: Proceedings of the 2019 IEEE/CVF international conference on computer vision. 2019- Octob; 2019. p. 1314–24. https://doi.org/10.1109/ICCV.2019.00140. IEEE.

Lichao Yang is a research fellow in Computer Vision and Artificial Intelligence at the School of Aerospace, Transport and Manufacturing at Cranfield University. He received his M.*Sc*. and Ph.D. degrees in automotive mechatronics and manufacturing from Cranfield University in 2018 and 2021. His-research interests include computer vision, image processing, machine learning, and human behaviour analysis.

Weixiang Du received the Ph.D. degree in manufacturing from Cranfield University, Cranfield, U.K., in 2021. He has been an Engineer working with Gansu Special Equipment Inspection and Testing Research Institute for about twelve years. His-research interests include the areas of computer vision, signal processing, NDT methods, and especially miniaturised NDT techniques for in-situ inspection.

Yifan Zhao was born in Zhejiang, China. He received the Ph.D. degree in automatic control and system engineering from the University of Sheffield, Sheffield, U.K., in 2007. He is currently a Reader in data science at Cranfield University, Cranfield, U.K. His-research interests include computer vision, signal processing, nondestructive testing, active thermography, and nonlinear system identification.

2023-07-06

A lightweight temporal attention-based convolution neural network for driver's activity recognition in edge

Yang, Lichao

Elsevier

Yang L, Du W, Zhao Y. (2023) A lightweight temporal attention-based convolution neural network for driver's activity recognition in edge. Computers and Electrical Engineering, Volume 110, September 2023, Article number 108861

https://doi.org/10.1016/j.compeleceng.2023.108861