Driver Monitoring with Deep Learning

1st Sheikh Muhammad Adib Bin Sh Abu Bakar
University of Applied Sciences Hamm-Lippstadt
Lippsatdt, Germany
sheikh-muhammad-adib.bin-sh-abu-bakar@stud.hshl.de

***Abstract*—Human error is one of the biggest contributors to car accidents today. For that reason, various kind of system have been built to reduce that error. One of the well-known system is called Advance Driver Assistance System (ADAS). Through a safe human-machine interface, ADAS increase car and road safety. Today, with advance research on computer neural network, ADAS has been improved with deep learning. This is the main theme of this paper, where driver monitoring system with deep learning will be discussed. The system is focus on detecting the drowsiness of the driver using deep learning so that appropriate action could be taken to ensure driver safety.**

***Index Terms*—ADAS, drowsiness detection, deep learning**

I. Introduction

car crash due to human accident [3]

drowsiness

the need of ADAS [6], [7]

driver monitoring

deep learning [1]

paper flow Drowsiness definition¿deep learning ¿ example application¿prosed method¿experiment¿result¿conclusion

II. Background

Before the method used for driver drowsiness detection is explained in detail, some basic terms used in this paper should be first comprehend so that the step used in the method's development can be clearly understood. Two things will be explained, drowsiness and neural network in deep learning, as an overview of the component of the proposed method.

A. Drowsiness

In this paper, a method used for drowsiness detection is proposed. Thus, it is important to have clear definition of drowsiness, so that the proper algorithm can be made. Drowsiness in this paper is defined when a driver closes their eyer longer than usual, which is 2 second. Means that, detecting the driver drowsiness of driver is measuring how long they close their eyes. This implementation will be further explained in proposed method section

B. Deep Learning

Deep Learning or known as deep structured learning [**?**], which are modelled after the structure and operation of the brain, are a subfield of machine learning approaches that deal with algorithms that use numerous layers to gradually extract higher-level properties from the raw input. Another frequently mentioned advantage of deep learning models, in addition to scalability, is their capacity for automatic feature extraction from unprocessed data, also known as feature learning [**?**]. A neural network is created to make prediction base on given input. That it, neural network is the core of deep learning. Just like human, before it could make any decision, it should learn how to make the right decision. This phase is called training phase after the neural network, or the model, is completely built. This phase will be detailed in the proposed method section. In this section, the basic building block of neural network will be explained to give an idea how neural network works. Neuron is the basic building block of neural network in deep learning. The neuron, depict in "Fig. 2", can have several input and output just like human neuron except this neuron, depict in "Fig. 1", is a set of mathematical function [**?**]. The multiple input can be visualized as shown in "Fig. 3".

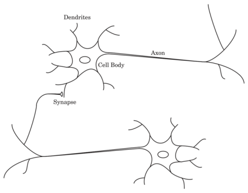

Fig. 1. Human neuron [1]

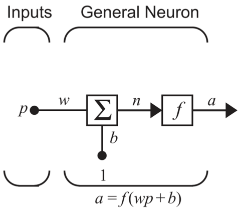

Fig. 2. Neuron with single input [1]

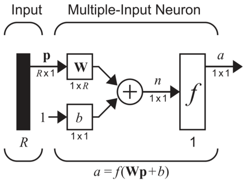

Fig. 3. Neuron with multiple input [1]

Every neuron has their own transfer function or activation function to satisfy some specification of the problem that the neuron is attempting to solve. “Fig. 4”shows the example of transfer function. Linear transfer function is used when the output should be linearly proportional to its input. Neurons with this transfer function are used in the ADALINE networks [**?**].

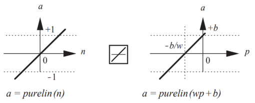

Fig. 4. Linear transfer function [1]

Another transfer function is log sigmoid, visualized in “Fig. 5”, which constricts the output to the range of 0 to 1 from the input that can have any value between plus and minus infinity. Because it is differentiable, the log-sigmoid transfer function is frequently employed in multilayer networks that are trained using the backpropagation process [**?**].

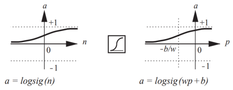

Fig. 5. Sigmoid transfer function [1]

The hard limit transfer function, visualized in “Fig. 6”, changes the neuron’s output from 1 to 0 depending on whether the function parameter is greater than or equal to 0. This process produces neurons that categorize inputs into two different groups. [**?**].

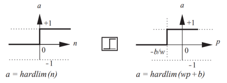

Fig. 6. Hard limit transfer function [1]

As mentioned before, the neuron is the basic building block of a neural network. This building block can be grouped into the form called layer as depict in “Fig. 7”. Every layer has their own functionalities and name. For instance, convolutional layer where the neuron acts as a kernel or filter, pooling layer, dropout layer and fully connected layers. These specific layers will be discussed more in the proposed method section.

The specific layers are then arranged in sequential way to form a neural network or network model. For example, Convolutional neural network (CNN) is widely used in neural network for image recognition that consist at least one layer of convolution layer [5]. Aforesaid, neuron in computer science is nothing more than a set of mathematical function that have input and output. That means a layer of neuron have a set of input and output as shown in “Fig. 7”. These variables are handled in matrix form where input matrix will be multiplied with weight and matrix of activation function and added with bias before supplied to the activation function. The output could be scalar or vector, depending on the input matrix and activation matrix. Since the input and output could be vector that have more than two dimensions, they are called tensor of input or output [5].

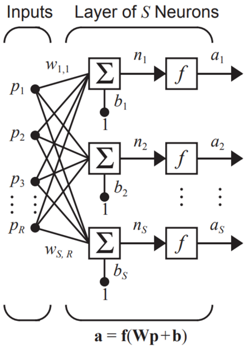

Fig. 7. A layer in neural network [1]

III. Related Works

Driver monitoring using deep learning is common these days. Many different method haven proposed. [4] for instance, use building block as describe in “Fig. 8” for their Driver monitoring system. Some blocks or modules depend on the output of others, for example, the input of the blink detection module depends on the output of the face detection module. The implementation of these building blocks makes use of CNNs, which has produced superior outcomes to more traditional image processing or computer vision-based approaches.

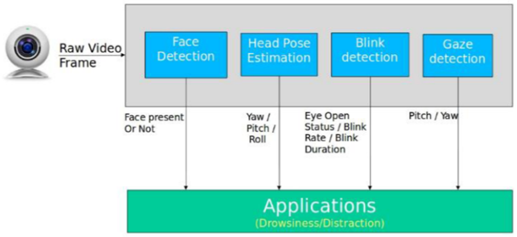

Fig. 8. Architecture used in [4]

A few steps have been taken to optimize the procedure because this model is intended to run on inexpensive embedded devices. The chosen platform is used for the training and inference phases. As a result, the platform can support the produced model’s complexity while maintaining the appropriate level of accuracy.

Pruning and quantization are also used to optimize the selected model. Although post-training methods like weight quantization are likely to produce results that are less accurate than the full precision version, the accuracy can be increased by retraining the network using the full precision version’s quantized weights as a starting point.

Utilizing more advanced deep learning techniques, such as replacing the convolution layer with a depth-wise convolution layer, is another way to reduce the model's complexity. Another method of optimizing for embedded platforms is choosing the appropriate deep learning technology. TensorFlow was chosen for this strategy because it supports remote computation and has effective graph visualization features, which are crucial, especially during training, and It offers C++ and Python APIs and compiles more quickly than most other libraries. Additionally, it includes the (Tflite) inference library, which is lightweight and designed for embedded devices.

This approach also supports multithreading, which helps make the most of multicore processors. Since some of the modules rely on the results of the earlier modules, this must be done strategically. This device is offered for sale and goes by the moniker See 'n Sense for the ARM/iMX platform.

[2] proposes a further deep learning-based solution for driver monitoring systems that focuses on drowsiness detection. This technique utilized a customized YoloV5 pretrained model and Vision Transformers (ViT) to analyse the robust binary image classification model, proposing a behavioural method framework from face detection to drowsiness detection. "Figure 9" depicts the framework. The dataset was increased through image augmentation.

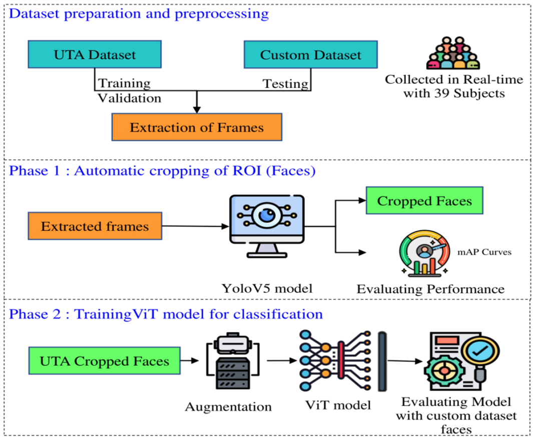
Dataset preparation and preprocessing
UTA Dataset
Custom Dataset
Training
Validation
Testing
Collected in Real-time with 39 Subjects
Extraction of Frames
Phase 1 : Automatic cropping of ROI (Faces)
Extracted frames
Cropped Faces
mAP Curves
YoloV5 model
Evaluating Performance
Phase 2 : TrainingViT model for classification
UTA Cropped Faces
Augmentation
ViT model
Evaluating Model with custom dataset faces

Fig. 9. Architecture used in [2]

After numerous thorough analyses, the YoloV5 obtained an accuracy rate of about 90%. The ViT model was evaluated, and the results showed that by using a custom dataset, the framework had achieved good average precision and sensitivity values.

The suggested architecture's drawback is that a substantial amount of data with labelled scene circumstances is needed in order to train the model. To reduce costs and increase computational efficiency without degrading, the design must be adapted for use in microcomputer systems. To improve the performance of the model, employ generative adversarial networks to expand the quantity of the training data.

In this paper, a method for drowsiness detection using deep learning is proposed that can improve some aspect from pervious method, especially in terms of accuracy. For those reasons, the method is also implementing transfer learning to increase the accuracy as implemented in [**?**] and use module technique as proposed by [4]. The dataset that will be used will be expanded using augmentation method as proposed by [2]. The detail about the method will be explained in proposed method section.

IV. Method

example [2], [4]
state of the art
proposed method [2]
component/architecture
step detail (from input to output)

In this method a simple drowsiness detection for driver is purposed using deep learning. This method consists of transfer learning where inceptionv3 model is used as a based model and combined with purposed neural network to decide either the eyes is close or not to define the state of the driver as discussed before. 1) base model The inceptionV3 model architecture is depict in "Fig **??**". The architecture within the box with dotted border is the part that are implemented in proposed method and it consist of 4 main component, which are convolution, maxpool (max pooling), avgpool(average pooling) and concat (Concatenet). These components are abstraction of layers of neuron or neural network [**?**]. The main objective of convolution is to extract features from the input image and produce feature maps. The output feature maps in the initial convolutional layer may learn to detect basic features, such as edges and colour composition variation [**?**]. By considering the input image as 3d matrix or tensor where hight, width and depth of the image is the parameter, the feature extraction is done by multiplying the matrix with another 3d matrix called filter or kernel as depict in "Fig **??**". The values in filter tensor are fixed in the way that a certain feature can be extracted, and the filter normally has a smaller size. So, the filter tensor will go through the image tensor by shifting the column and row of the image tensor. The step of shifting is formally called stride. Since the result of multiplication of a matrix will produce smaller matrix, the resulting image also have the smaller tensor. However, some convolution layer can maintain the size of input tensor by expending the input tensor. This process is known as padding. [**?**] Pooling is a function where the spatial size of the representation is reduced to reduce the amount of parameter and computation in the network. Pooling layer operates on each feature map independently. There are two type of pooling which are max pooling and average pooling [**?**]. Max pooling is operated by selecting a region with a fixed size in input tensor and select the highest value to form a new tensor as an output. The region than move by the step of fixed stride. As a result, smaller tensor is produced. This process is depict in "Fig **??**". In average pooling layer, the output is the average value of selected region. Concatenate layer is the layer that combine more than one input tensor to become one output tensor. This out put will be use by the next layer that require tensor that are bigger than one of the input tensors sizes. 2) Purposed network The purposed neural network is visualized in "Fig **??**" where the input from the inceptionV3 model is flatten. Since the final output is only binary decision because only the state of the eyes need to be detected, either they are close or not, the output from flatten layer is reduce to 64 neuron as an input to dropout layer to reduce the number of output into two output. Activation function Relu Softmax

V. Experiment

concept

flow

component

dataset

training phase

As discussed before, the neuron has two main parameter, namely weight and bias. These parameters are vital to produce the right output, so that a precise decision can be made. The method that is used to find the right value for the parameter is called training and the data that are used for training is called data set. The model is then tested at the end of training phase or known as inference to evaluate the model.

1) Work Flow: ¡¡¡Before discussing the detail about the model use case and implantation, the overall workflow will be explained first. The model will be implemented in two stage, training, and application. In the training phase, the model's parameter is adjusted, so that the detection accuracy can be improved. In application stage, the model will use the trained parameter to make prediction in a real environment. In the next section, the detail about training stage will be discussed further.¿¿¿

2) Data Set: Data sets are used as an input for the model to enable it to be train. The output that produce will be evaluated. Datasets are usually grouped into batches to handle huge number of data. Some people use the term iteration loosely and refer to putting one batch through the model as an iteration.

3) Training: training a network means nothing more than solving a complex optimization problem [**?**]. At first, the value of weight and bias for every neuron is randomly assigned. After that, an image is given to the model as an input. The final output value is compared with an expected value, where the difference value or known as loss value is recorded. The loss values are used by loss function to improve the value of the weight in the way that the loss value can be reduced in the future. The lost function that is used in proposed method is categorical cross entropy. Since the provided dataset for the proposed method is very large, having more than one epoch is necessary. An epoch indicates the number of passes of the entire training dataset the model has completed. The general relation where dataset size is d, number of epochs is e, number of iterations is i, and batch size is b would be d*e = i*b. [**?**] Determining how many epochs a model should run to train is based on many parameters related to both the data itself and the goal of the model.

4) Inference: During the training phase, not all data from the dataset are used for training, some of them al also used to evaluate the model. This process is known as inference. For instance, 80

5) Platform: The proposed method is design using python environment and the API used for the model are from TensorFlow[1]. The training phase was executed on computer with 64-bit operating system, x64-based processor, Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz 1.50 GHz and 16 GB RAM.

VI. Result

test phase

result

VII. Conclusion

deep learning - help monitor driver - improve adas - reduce car crash

Acknowledgment

References

[1] Martin T. Hagan, Howard B. Demuth, Mark Hudson Beale, and Orlando De Jésus. *Neural network design*. Martin T. Hagan, 2nd edition edition.

[2] Ghanta Sai Krishna, Kundrapu Supriya, Jai Vardhan, and Mallikharjuna Rao K. Vision transformers and YoloV5 based driver drowsiness detection framework. Publisher: arXiv Version Number: 1.

[3] David Matine. What percentage of car accidents are caused by human error? | pittsburgh law blog.

[4] Nirmal Kumar Sancheti, Manjari Srikant, and Krupa H Gopal. Camera based driver monitoring system using deep learning. page 5.

[5] Mohit Sewak, Md Rezaul Karim, and Pradeep Pujari. *Practical Convolutional Neural Networks: Implement advanced deep learning models using Python*. Packt Publishing.

[6] Aleksandra Simic, Ognjen Kocic, Milan Z. Bjelica, and Milena Milosevic. Driver monitoring algorithm for advanced driver assistance systems. In *2016 24th Telecommunications Forum (TELFOR)*, pages 1–4. IEEE.

[7] Lishengsa Yue, Mohamed A. Abdel-Aty, Yina Wu, and Ahmed Farid. The practical effectiveness of advanced driver assistance systems at different roadway facilities: System limitation, adoption, and usage. 21(9):3859–3870.

[1]