TEST Engineering & Management

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

Research on End to End Control Autonomous Driving System Based on Deep Learning

Guobao Xu[1,a *], Zhenjian Zhu[1,b], Hongwei Li[2,c], Ji Wang[1,d *]

[1] School of Electronics and Information Engineering, Guangdong Ocean University, Zhanjiang 524088, China

[2] Institute for Chemicals and Fuels from Alternative Resources (ICFAR), Western University, London, Ontario N6A 5B9, Canada

[a] xuguobao@126.com, [b] zhuzhenjian2@163.com, [c] lihongweinvahai@163.com, [d] zjouwangji@163.com

Article Info
Volume 83
Page Number: 574 - 585
Publication Issue:
July-August 2020

Article History
Article Received: *06 June 2020*
Revised: *29 June 2020*
Accepted: *14 July 2020*
Publication: *25 July 2020*

Abstract

Deep learning machine algorithm is successfully used in automatically driving of a toy car. Firstly, the car was controlled by the system wirelessly to move along the designated route, and the frame sample data and simultaneously records of the car direction operation were collected using a camera to make the training data and label separately. Secondly, the ResNet convolutional neural network was established by using the frame sample data to predict the operation direction of the car. Thirdly, the loss value calculated from the predicted action was compared with the value from actual operation, and reduced by implementing the gradient descent algorithm. After a series of experimental processes, the predicted value of the system can be equivalent to the value obtained from actual driving action. The experimental results show that the processing rate can effectively reach more than 30 fps with a high accuracy.

Keywords: *Autopilot, Deep Learning, Neural Networks, Gradient Descent, Prediction*

1. Introduction

With the continuous development of artificial intelligence, more and more attention has been paid on the research of autonomous driverless cars. The system used in smart car is a highly computer-based and complexly integrated, which is composed of environmental awareness, planning and decision making, and multi-level assisted driving. It is well known that the transportation mode in the future will be significantly changed, and probably replaced by driverless cars. The research on self-driving started by Google and then attracted numerous attentions from all over the world. In 2009, Google began to test its own self-driving car, which has been tested for 2.89 million km untill July 2016 [1].However, high accuracy of lane detection, obstacle detection and traffic identification of self-driving car cannot be achieved without the deep learning algorithm [2-7].

Nowadays, there are three main methods to realize automatic driving based on computer vision, i.e., indirect perception, direct perception and end-to-end control. Indirect perception system is a traditional auto-driving system, including four sub-modules of target tracking, target detection, camera model and calibration, and 3d reconstruction[9].This method is realized to integrate the detection results from the above modules to establish a complete environment

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

representation. Researches on this system is very abundant, resulting in the improvement of technologies in self-driving cars. However, the system has obvious disadvantages with high complexity and redundancy, and high cost for commercial application.

Direct perception system is a system improved based on the indirect perception system [10]. The learning artificial indicators are used to describe the traffic scene without the dependence on each sub-module. For example, when the auto-vehicle is detecting the distance between itself and other nearby vehicles on the road, for the traditional indirect perception system, the target vehicle should be detected by standard first, and then converted into distance through the camera model calculation, while the direct perception system is to learn the vehicles distance by neural network, leading to a lower system complexity. However, this method is relatively weak in adaptability and high in specificity, and specific traffic scenes are needed to maximize its advantages. It may also be difficult to migrate to other scenes.

The automatic driving system using end-to-end control system directly uses the neural network training learning method to learn the driving action, which does not divide the system into multiple sub-modules to perceive the driving environment [11]. The visual image is input in the system according to the trained neural network, and then the system will judge the current corresponding left-turn, right-turn or straight-line driving actions according to the predicted results of the network output, which can be attributed to the image classification. As the number of layers of neural network increases, the abstraction layer of feature becomes higher, and the approximation effect of the system function becomes better. This system has a low complexity, and easy to obtain data, and more suitable for small scenarios. Pomerleau [12] successfully established a set of autonomous land vehicle in a neural network (ALVINN) in 1989, which proved the feasibility of end-to-end training neural network model for auto-driving.

At the algorithm level, the automatic driving system also relies heavily on research level of deep learning and computer vision technologies. In recent years, deep learning and computer vision technologies have developed rapidly, but it still takes a long time to realize automatic driving in any complex environment. Among the three common automatic driving methods, the indirect perception method is more traditional with high system complexity, a large amount of computing resources and often high investment cost. However, the research of the other two system methods is still in the early stage, and the extensibility and stability still need to be verified by long-term theoretical research. The method described in this research belongs to the end-to-end control method, which is significantly important for the system development.

2. An Overview of Convolutional Neural Network Organization of the Text

Convolutional neural network (CNN) belongs to the category of artificial neural network [13]. Because its network model is usually composed of multiple layers, it is also called deep convolutional neural network. It is one of the typical models of deep learning. Compared with the traditional artificial neural network, convolutional neural network is local link and weight sharing, which reduces not only the weight number of the network but also the complexity of the network model. Therefore, the network is suitable for tasks with abundant information and large amount of input data, e.g., image recognition. This is also the reason why this research chooses convolutional neural network as the basic network model. A convolutional neural network generally includes five layers neural network structures, i.e., input layer, convolutional layer, pooling layer, full connection layer, and Softmax layer, as shown in Fig.1.

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

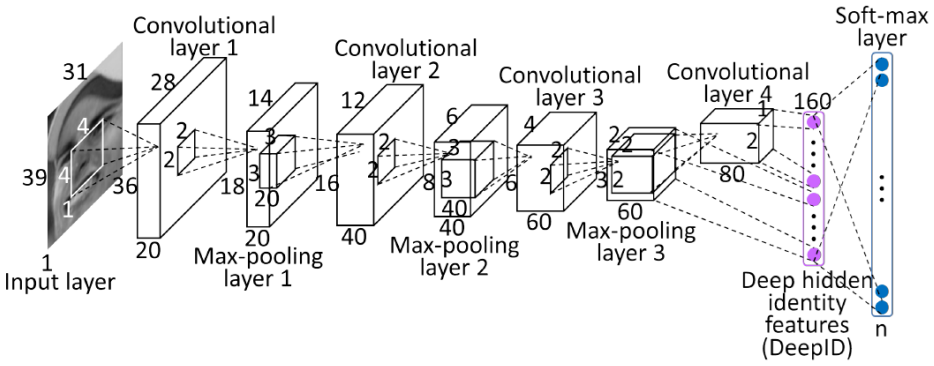

Fig. 1. Structure of Convolutional Neural Network

2.1 Convolutional Layer

The convolution layer, called as filter, is the main part of the convolutional neural network structure [14]. The filter is used to extract the characteristic information of the input data by performing convolution calculation in the local sensing domain. Different filters can extract different features, such as shadows, outlines and so on. With the characteristic of weight sharing, it can greatly reduce the training parameters on neural network. The convolution operation is the inner product operation (element multiplication and then summation) between the image local window data and the filtering matrix, as shown in Fig. 2. The left side represents the original input data, the middle part is the filtering matrix, and the right side is the two-dimensional data obtained after the convolution operation.

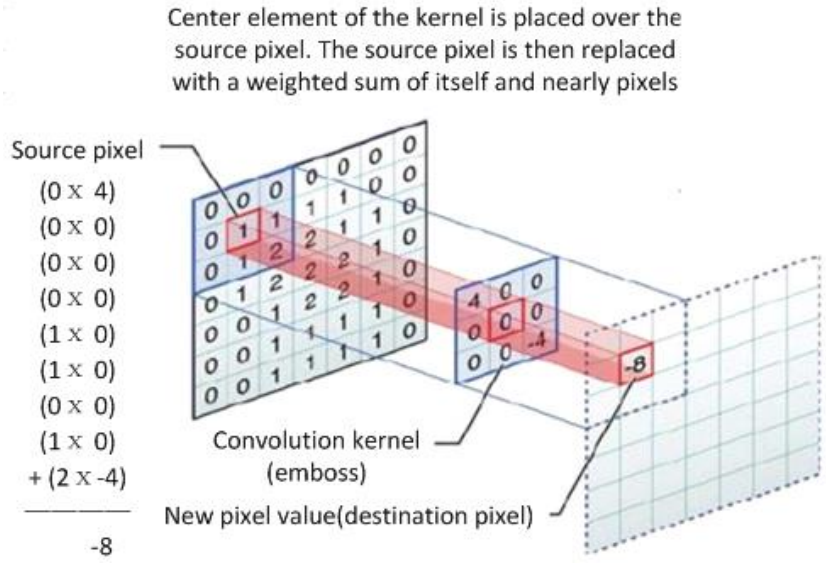

Fig.2. Convolution Procedure

In the process of CNN training, multiple convolution cores are often used for convolution of the input data [15]. After the data window convolution operation is completed, the data window will be shifted and moved until all data is traversed, and then the next convolution operation will be processed. Each convolution kernel generates a layer feature graph. The mathematical expression of the above process is shown in Eq. 1.

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

$$a_{i,j} = f(\sum_{d=0}^{D-1}\sum_{m=0}^{F-1}\sum_{n=0}^{F-1} w_{d,m,n} x_{d,i+m,j+n} + w_b) \quad (1)$$

Where, $x_{d,i+m,j+n}$ represents the element of the input image in the i th row and jth column of the dth layer, $w_{d,m,n}$ represents the weight value of the convolution kernel in the mth row and nth column of the dth layer, d represents the depth. F represents the width and height information of the convolution kernel, w_b represents the partial term of the convolution kernel, $a_{i,j}$ represents the element of the feature graph in the ith row and jth column, and $f()$ represents the activation function.

2.2 Pooling Layer

The pooling layer is inserted between two successive convolution layers. Pooling layer is commonly recognized as the conversion of one image with high resolution into another image with low resolution. It is used to reduce sampling, the spatial size of data blocks in the network gradually, and the number of parameters in the network, while it improves the iteration rate of model learning, and overcome overfitting [16].

Similar to the convolution layer, the forward propagation process of the pooling layer is also performed by moving the structure of similar filters. However, the calculation of the pooling layer filter is not the weighted sum of nodes, but simply the maximum value. The pooling layer that uses the maximum operation is called the maximum pooling layer, and the pooling layer of this structure is used frequently, as shown in Fig.3. The filter size is 2 x 2, and the pooled step size is 2, which means that the input data of 4 x 4 size is changed to 2 x 2 through the pooling layer.

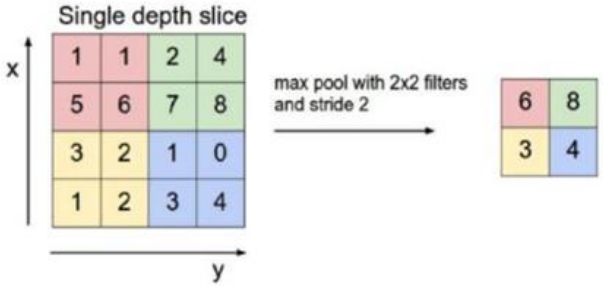

Fig. 3. Maximum Pooling Layer Algorithm

2.3 Loss Function

Cross entropy is one of the commonly used loss functions to evaluate the gap between predicted value and expected value.The specific formula of cross entropy is shown in Eq. 2, where p and q are respectively the probability distributions of prediction and expectation.

$$H(p,q) = -\sum_{x} p(x)\log q(x) \quad (2)$$

According to the above formula, cross entropy describes the difference between the probability distribution of prediction and expectation, but in fact, the output of neural network is not necessary to the probability distribution, which describes the probability of different events. The probability of any event has a probability between 0 and 1, and there will always be an event to happen. To solve the problem of turning the output of neural network into a probability distribution, Softmax regression method is used. Suppose that the output prediction vectors of the neural network are $y_1,y_2,...,y_n$, then the output after using Softmax regression method is:

$$soft\max(y)_i = y'_i = \frac{e^{yi}}{\sum_{j=1}^{n} e^{yj}} \quad (3)$$

According to the above formula, the output of the original neural network is used as confidence to generate new output, which meets all requirements of probability distribution. In this way, the output of the neural network is turned into a probability distribution, so that the distance between the predicted probability distribution and the actual probability distribution of the real answer can be

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

calculated by cross entropy.

In this research, the probability distribution of left-turn, right-turn and straight-line driving actions predicted by CNN is a multi-classification problem, so the loss function uses cross entropy.

2.4 Neural Network Optimization Algorithm

The function of the optimization algorithm is to minimize (or maximize) the loss function J (x) by improving the training method. Some parameters in the model are used to calculate the deviation between the real and predicted values of the target value Y in the test set. Based on these parameters, the loss function J (x) is formed. For example, weight (W) and deviation (b) are internal parameters commonly used to calculate output values and play a major role in training neural network models. The internal parameters of the model play an important role in effectively training the model and producing accurate results. This is the reason why various optimization strategies and algorithms are used to update and calculate network parameters that affect model training and output to make them close to or reach the optimal value.

Reverse propagation algorithm and gradient descent algorithm are the core optimization algorithms of neural network. Gradient descent algorithm is to optimize the single parameter in the network to reduce the loss function. The reverse propagation algorithm performs gradient descent on all parameters of the entire neural network, optimizes all parameters on the defined loss function, and makes the overall loss function of the neural network reach a small value.

Adam algorithm is an adaptive time estimation method, which can further optimize gradient descent and has a good acceleration effect on deep network training. The algorithm can calculate the adaptive learning rate of each parameter, which not only stores the exponential decay mean of the previous square gradient of AdaDelta, but also retains the exponential decay mean of the previous gradient m_t.

m_t is the mean value at the first moment of the gradient, and v_t is the non-central variance value at the second moment of the gradient.

$$\hat{m}_t = \frac{m_t}{1-\beta_2^t}$$
$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \quad (4)$$

Then, the final formula for parameter update is:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \varepsilon}}\hat{m}_t \quad (5)$$

Where, $\beta_1$is set to 0.9, β_2 is set to 0.9999, and ϵ is set to10^{-8}.

In practical application, Adam algorithm works well. Compared with other adaptive learning rate algorithms, Adam algorithm is more effective and its convergence speed is faster. In addition, some problems in other optimization techniques can be corrected, such as too slow convergence speed, disappearance of learning rate or parameter update with high variance, so that the loss function fluctuates greatly.

3. Structure Design of Convolution Network Model

There are many kinds of neural network models that can be obtained by combining the five network structures of input layer, convolutional layer, pooling layer, full connection layer and Softmax layer, among which many excellent classical convolutional network models are proposed to solve image processing problems with better effects, such as VGGNet and ResNet. In deep network training, ResNet has more obvious advantages. According to the latest recommendation ranking of deep learning network model, ResNet has become one of the most recognized efficient neural network models. The proposed deep learning algorithm adopts the ResNet network model to control the training network of the car in this research.

ResNet's method aims at solving the problem of deep neural network degradation. As the depth of the neural network increases, the accuracy continues to rise until it reaches saturation. If the depth continues

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

to increase, the accuracy will decrease. This situation is not caused by over-fitting, but because in the process of neural network training, not only the error of test set increases, but also the error of training set increases. Assuming that the network has reached the accuracy of saturation, a pair of y = x complete mapping layers are added later, and the error will not rise, that is, the purpose is achieved by adding residual network blocks, as shown in Figure 4. Deeper neural networks should not lead to increased error. The full mapping layer addition principle mentioned just now is the source of inspiration for ResNet. It worths mentioning that, ResNet won the ILSVR championship in 2015 and reached the top5 error rate of 3.57%.

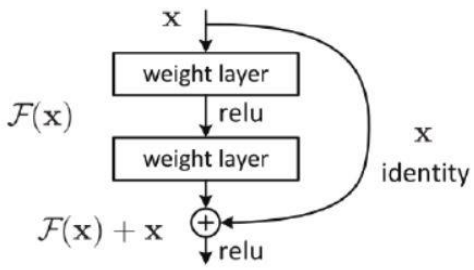

Fig.4. ResNet Residual Network Block

After further theoretical research verification and development, ResNet generally has 18 layers, 34 layers, 50 layers, 101 layers, and 152 layers depths of the superior structure. ResNet-50 and ResNet-101 are particularly commonly used. The ResNet-50 convolutional neural network model is adopted in this research. As shown in Fig.5., the ResNet module is a partial incomplete block of ResNet-50 (including three-layer convolutional network), which is known as the "building block". According to many relevant researches, the ResNet-50 network is generally divided into five parts, namely conv1, conv2_x, conv3_x, conv4_x and conv5_x. Each part contains multiple "building blocks", and the network in all "building blocks" has 50 layers in total. Finally, image classification is processed through the full connection layer and Softmax layer. The ResNet-50 convolutional neural network model is implemented in TensorFlow framework. TensorFlow is a mainstream deep learning framework of Google company open source, supports a variety of deep learning algorithms, and can automatically complete calculus and other complex calculations. In addition, TensorFlow provides the interface to support C++ and Python language, at the same time cross-platform operation. TensorFlow uses data flow diagram to describe the calculation process, and converts each calculation into a node on the calculation diagram, and the lines between nodes represent the connections between calculations.

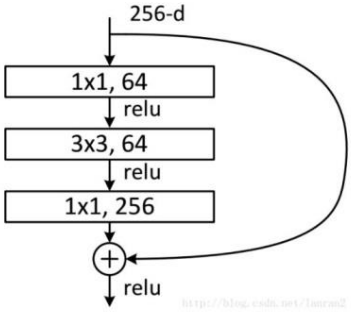

Fig.5. ResNet-50 Incomplete Block

4. System Design and Implementation

This design adopts the toy car carrying Raspberries pie as experiment platform, and the car was driven by using an USB camera and drive motor. The image data collected by the camera was transmitted by using the Raspberry pie to the remote host via wireless network, and then the driving prediction results was returned by the remote host to the Raspberry pie for execution. This design is implemented by TensorFlow Google deep learning framework. The 320*240 resolution image data collected by the car in real time were taken as the input of convolutional neural network, and the driving action of the same timestamp was taken as the data label, that is, each training data should contain the first visual image of the car and the appropriate driving direction corresponding to the image. The first visual image of the car and the corresponding driving action were all manually

selected. Therefore, this algorithm was the supervised learning algorithm and the sample collector was the supervisor. In addition, in order to ensure the representativeness and adaptability of training samples, data was collected under different light intensity and environmental sites. And the consistency of the data acquisition strategy greatly affected the entire network study effect. The car data acquisition was focused on the first representative of visual images, such as the car in front of barrier-free was kept driving direction, when close to the obstacle began to adjust steering, which is more conducive to image classification. The data acquisition should also try to reduce the occurrence of bad data, to prevent the overfitting caused by noise data. The sample data was divided into 80% training set and 20% test set, and the training set was divided into small batch set again, which was trained by Adam adaptive learning rate optimization algorithm. The network output the probability distribution of forward, the left and right, and cross entropy was used as the loss function. By continuously reducing the loss function and updating the weight of network training parameters, the optimal values of all training parameters were achieved. In contrast, an optimal abstract feature sampler was trained in the convolution layer. After one iteration of training, the test set was used for verification to calculate the test set accuracy. According to the high test set accuracy, a neural network model with good effect was finally obtained.

4.1 Hardware Design

According to the functional requirements of the automatic driving system based on end-to-end control, the hardware was mainly divided into four parts: motor drive design, camera design, Raspberry pi hardware platform, and remote host PC terminal interaction. The overall design framework is shown in Fig.6.

DC motor driver chip L298N: receive Raspberry pie and send PWM signal to control motor rotation.

Raspberry pi 3B main control: collect camera data and send frame data to PC through wireless wifi; Accept the PC terminal driving action instructions to drive the motor.

USB camera: collect real-time image data when the car is moving.

Remote host: receive the frame data of Raspberry pie and calculate the prediction result of convolutional neural network model, return the driving command to Raspberry pie.

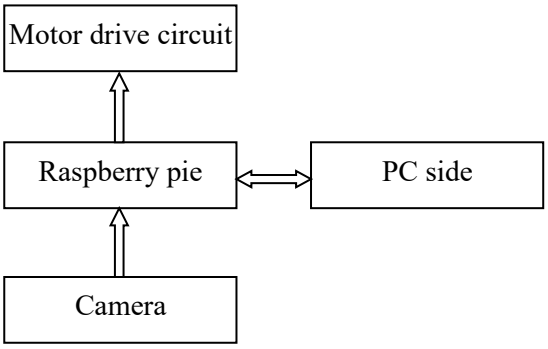

Fig.6. Hardware Design Block Diagram

4.2 Software Design

Firstly, system logged in Raspberry pie remotely through network protocol SSH. Then, make sure that the time zone and time synchronization between Raspberry pie and PC before data sample collection began. It then called the camera in the Raspberry pie using the ffmpeg tool and stored the video buffer stream. It started the program script to receive the driving command of the car, and created a new command receiver web application service in the Raspberry pie, which is responsible for receiving and executing the driving action on the wireless network at any time.

The PC side read the image frame data from ffmpeg server service port of Raspberry pie in real time, and used OpenCV to open the video stream on the PC side. While obtaining the camera angle of Raspberry pie, the access command received the web service. Raspberry pie received the PC keyboard key value from the port. According to the analysis of key values, it is to find the corresponding driving

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

command and to achieve the computer remote synchronous control of the car. Fig.7. and Fig.8. respectively show the flow chart of PC and Raspberry pie.

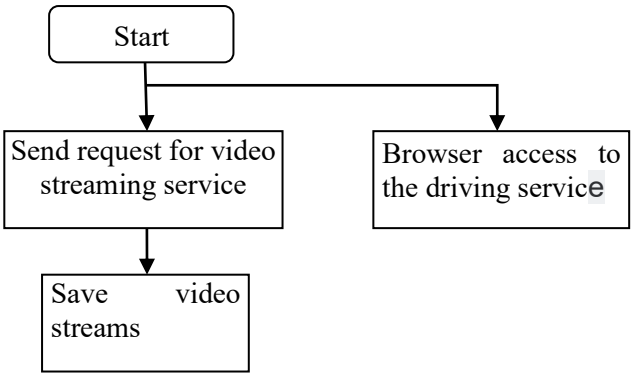

Fig.7. Sample Data Collection Software Design Framework of PC

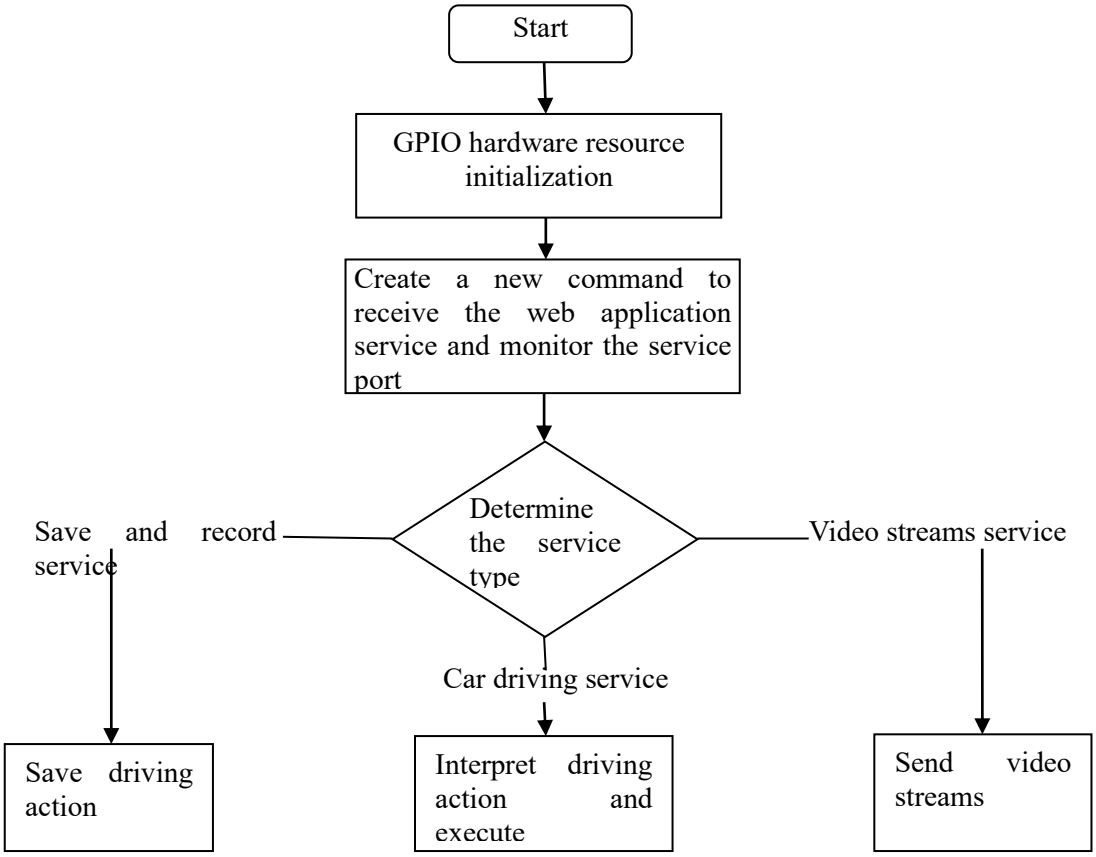

Fig.8. Sample Data Collection Software Design Framework of Raspberry Pie

4.3 System Test

The site of the software collection was in room 04010, zhonghailou laboratory, Guangdong Ocean University. Firstly, the background web service released by Raspberry PI was accessed on the remote host terminal, and then the keyboard key value was transmitted through the wireless network to control the car movement and record the time stamp of the driving action. Meanwhile, the Raspberry PI in the car sent the camera data to the host for video streaming data storage. Finally, the timestamps of the video stream and driving operations were matched and packaged into sample data. Fig. 9 shows data acquisition process.

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

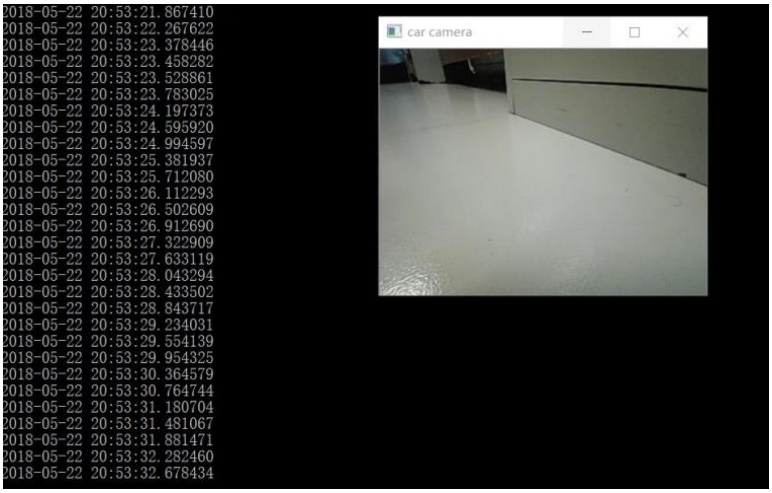

Fig.9. Data Collection Process

The framework tool of TensorFlow was used to train ResNet-50 neural network model, in which the system selected the graphics card platform NVIDIA GTX1050, collected about 2GB image data volume, and iterated training for 100 times in the convolutional neural network, and the GPU consumed about 30 hours in training. After the input of sample data, the network model kept learning and training. With the increase of the number of iterations, the accuracy became higher and higher, up to 89%.The experimental results show that the frame rate can reach above 30 fps with high accuracy. Fig. 10 shows the training partial results of the convolutional neural network.

```
epoch: 0, training accuracy: 0.3499999940395355, validation accuracy: 0.44999998807907104
epoch: 1, training accuracy: 0.5, validation accuracy: 0.5249999761581421
epoch: 2, training accuracy: 0.5, validation accuracy: 0.4749999940395355
epoch: 3, training accuracy: 0.574999988079071, validation accuracy: 0.6499999761581421
epoch: 4, training accuracy: 0.4749999940395355, validation accuracy: 0.5249999761581421
epoch: 5, training accuracy: 0.5, validation accuracy: 0.4000000059604645
epoch: 6, training accuracy: 0.574999988079071, validation accuracy: 0.5
epoch: 7, training accuracy: 0.5, validation accuracy: 0.6000000238418579
epoch: 8, training accuracy: 0.675000011920929, validation accuracy: 0.574999988079071
epoch: 9, training accuracy: 0.6000000238418579, validation accuracy: 0.675000011920929
epoch: 10, training accuracy: 0.6000000238418579, validation accuracy: 0.574999988079071
epoch: 11, training accuracy: 0.675000011920929, validation accuracy: 0.550000011920929
epoch: 12, training accuracy: 0.699999988079071, validation accuracy: 0.824999988079071
epoch: 13, training accuracy: 0.699999988079071, validation accuracy: 0.699999988079071
epoch: 10, training accuracy: 0.574999988079071, validation accuracy: 0.5
epoch: 11, training accuracy: 0.625, validation accuracy: 0.550000011920929
epoch: 12, training accuracy: 0.75, validation accuracy: 0.625
epoch: 13, training accuracy: 0.625, validation accuracy: 0.6000000238418579
```

Fig.10. Partial Results of Neural Network Training

July-August 2020
ISSN: 0193-4120 Page No. 881 - 891

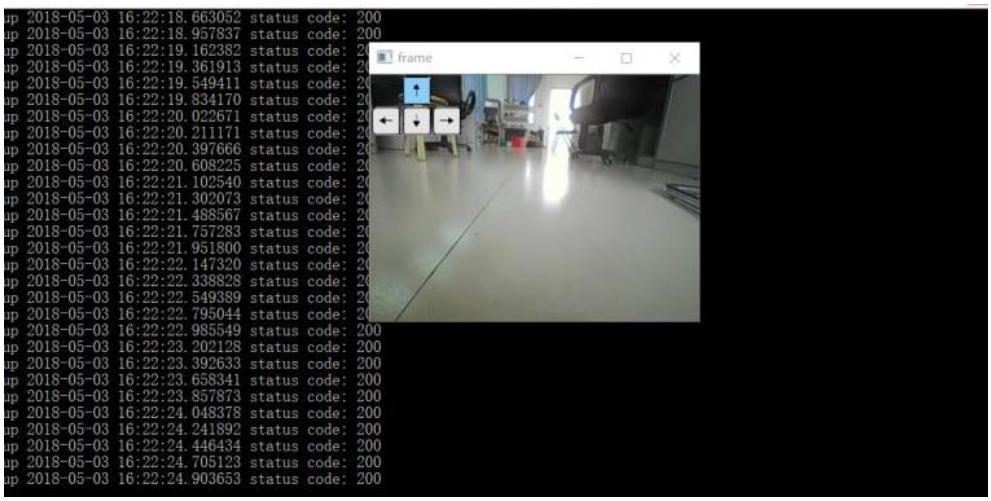

Fig.11. Test Result Using Neural Network

5. Conclusions

In this research, an end-to-end driving scheme model based on deep learning was designed and implemented. The real-time traffic images were learned through the convolutional neural network to map out the car driving actions. This research firstly introduced the deep learning algorithm, namely the deep convolutional neural network algorithm, and described the structure model, principle and method of the convolutional neural network. Secondly, the structure of the convolutional network model was presented in detail. Thirdly, the Google TensorFlow deep learning open source framework was used to train algorithm. Experimental results verified the feasibility of end-to-end autonomous driving technology. The defect of this system is that the sensor is not perfect and the system is limited by the uncomplicated field. The overall design of the system needs to be further improved with correct theoretical support, while the deep learning algorithm theory needs to be developed over a longer period of time.

Acknowledgement

This work was financially supported by 2018 Guangdong Province Key Platform and Major Scientific Research Projects -- Characteristic Innovation Projects (Grant Nos. 2018GXJK065), Major Scientific Research Project Training Program of Guangdong Provincial Department of Education (GDOU2017052602) and 2018 Guangdong Engineering Technology Research Center (Grant Nos.〔2018〕2580), 2017 Provincial College Students Innovation and Entrepreneurship Training Program (Grant Nos. CXXL2017079)

References

1. X. Chen. The Study on the Challenge and Development Prospect of Automated Vehicles. CHINA TRANSPORTATION REVIEW, 2016, (38): 9-13.
2. J. Long, E. Shelhamer, T. Darrell. Fully Convolutional Networks for Semantic Segmentation, Proc of IEEE Conference on Computer Vision and Pattern Recognition, 2015: 3431-3440.
3. K. Simonyan, A. Zisserman. Very Deep Convolutional Networks for Large-scale Image Recognition. Computer Science, 2014:1409+1556.
4. V. Badrinarayanan, A. Kendall, R. Cipolla. Segnet: a Deep Convolutional Encoder-decoder Architecture for Image Segmentation. IEEE Trans on Pattern Analysis and Machine Intelligence, 2017, (39); 2481-2495.
5. S. Ren , K. He, R. Girshick, et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE

May-June 2020
ISSN: 0193-4120 Page No. 1 - 08

Transactions on Pattern Analysis & Machine Intelligence, 2015, 39(6): 1137-1149.
6. Krizhevsky, I. Sutskever, G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks, Proc of International Conference on Neural Information Processing Systems, 2012: 1097-1105.
7. H.C. Shin, H.R. Roth, M. Gao, et al. Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. IEEE Transactions on Medical Imaging, 2016: 1-1.
8. Szegedy, W. Liu, Y. Jia, et al. Going Deeper with Convolutions, Proc of IEEE Conference on Computer Vision and Pattern Recognition, 2015: 1-9.
9. S. Ullman. Against Direct Perception, Behavioral & Brain Sciences, 1980, 3(3): 373-381.
10. Chen, A. Seff, A. Kornhauser, et al. Deep Driving: Learning Affordance for Direct Perception in Autonomous Driving, 2015 IEEE International Conference on Computer Vision(ICCV), 2015: 2722-2730.
11. Y. Lecun, U. Muller, J. Ben, et al. Off-road Obstacle Avoidance through End-to-end Learning, International Conference on Neural Information Processing Systems, 2005: 739-746.
12. A. POMERLEAU. ALVINN: an Autonomous Land Vehicle in a Neural Network, Advances in neural information processing systems 1. Morgan Kaufmann Publishers Inc. 1989.
13. X. Han. Deep Learning Based Scene Recognition for Autonomous Driving, Sun Yat-Sen University, 2017.
14. Z. Yang. Research on Computer Vision Feature Representation and Learning for Image Classification and Recognition, South China University of Technology, 2014.
15. Y. Li , Z. Hao , H. Lei. Survey of Convolutional Neural Network, Journal of Computer Applications, 2016, (36): 2508-2515+2565.
16. Z. Sun, Chengxing Lu, Zhongzhi Shi, Gang Ma. Research and advances on Deep Learning, Computer Science, 2016, (43): 1-8.