Neural Comput & Applic
DOI 10.1007/s00521-016-2277-9

ORIGINAL ARTICLE

Moving object recognition using multi-view three-dimensional convolutional neural networks

Tao He[1] · Hua Mao[1] · Zhang Yi[1]

Received: 18 January 2016 / Accepted: 2 March 2016
© The Natural Computing Applications Forum 2016

Abstract Moving object recognition (MOR) is an important but challenging problem in the field of computer vision. The aim of MOR is to recognize moving objects in a given video dataset. Convolutional neural networks (CNNs) have been extensively used for image recognition and video analysis problems. Recently, a 3D-CNN, which contains 3D convolution layers, was proposed to address MOR problems by successfully extracting spatiotemporal features. In this paper, a multi-view (MV) 3D-CNN is proposed for MOR. This model combines 3D-CNNs with a well-known MV learning technique. Because multi-view learning techniques have the ability to obtain more view-related features from videos captured by different cameras, the proposed model can extract more representative features. Moreover, the model contains a special view-pooling layer that can fuse the feature information from previous layers. The proposed MV3D-CNN is applied to both real-world moving vehicle recognition and sign language recognition tasks. The experimental results show that the proposed model possesses good performance.

Keywords Moving object recognition · Multi-view learning · 3D convolutional neural networks · Feature extraction · Deep learning

✉ Hua Mao
huamao@scu.edu.cn

Tao He
taohe@stu.scu.edu.cn

Zhang Yi
zhangyi@scu.edu.cn

[1] Machine Intelligence Laboratory, College of Computer Science, Sichuan University, Chengdu 610065, People's Republic of China

1 Introduction

One of the fundamental challenges in computer vision concerns moving object recognition (MOR), which attempts to recognize moving objects within video data. Typical MOR tasks include human motion recognition [1, 2], action recognition [3, 4], and object tracking [5]. Most MOR methods are based on the analysis of images captured from a given video dataset. The basic method of solving MOR problems is to recognize moving objects using machine learning classifiers. As spliced image frames captured from the video cannot be directly used as the input to these classifiers, the general solutions of MOR problems start with certain preprocessing methods. These methods, which are performed on the spliced image frames, extract features to represent each image frame. In [6], the bag of words (BoW) and SIFT-based methods are used to extract features from soccer videos.

Convolutional neural networks (CNNs), which are typical deep learning models, have been widely used for image recognition problems such as CIFAR, ImageNet, and large-scale visual classification missions [7, 8]. CNNs contain a hierarchy of convolution and pooling layers and can automatically learn potential features from a training dataset to achieve better performance than hand-crafted features. Another advantage is that CNNs have fewer connections, which means the training stage is faster than in some other neural networks [9]. Because of their feature learning ability, CNNs are also used to solve MOR tasks [4, 10, 11]. In [4], authors tried to learn the spatiotemporal features of videos by a parallel CNN architecture which is based on both spatial and temporal streams. The previous one focuses on recognizing objects from continuous video frames, and the later one is used to recognize action from motion in the form of the dense optimal flow stream. Two

Published online: 23 March 2016

streams are independent of each other and combined by late fusion. In [10], a multi-resolution CNN architecture is proposed to train on large-scale datasets. In [11], a series of fusion strategies are applied in general CNN models to solve MOR problems.

However, general CNNs are limited to learning spatial features from a single image. As a variant of CNNs, 3D-CNNs are designed for learning the features of a video directly. The 3D-CNN [12] treats the temporal axis in the same manner as other spatial axes to extend the ability of convolution layers to extract spatiotemporal features. In [3], 3D convolution layers are applied over video clips to learn object features from image frames, respectively, and then fuse recognition results at the video level. The 3D-CNNs applied in human action recognition tasks have demonstrated good performance.

In MOR tasks, the video data usually consist of videos captured from cameras with different views. For example, in many video surveillance systems, there are a number of cameras placed at different shooting angles to prevent the control space from being exposed. The view information of a moving object is considered an important factor in MOR tasks. Multi-view learning [13] is an interesting visual feature learning domain in which datasets consist of diverse feature subsets called multiple views. Recently, there has been an attempt to combine multi-view learning and CNNs in the field of static object recognition [14]. The authors employed multiple CNNs to compile information from multiple views of static objects into a compact shape descriptor of the static object, offering better recognition performance than single-view methods. This method regards different shapes extracted from static objects as distinctive views.

Further extending the idea of 3D-CNNs and multi-view learning, we propose a multi-view 3D convolutional neural network (MV3D-CNN) architecture for addressing MOR problems. We consider using 3D-CNNs as a single video view descriptor. Our model aims to extract enough view information from videos using different 3D-CNN descriptors. The main contributions can be summarized as follows:

- A set of parallel 3D-CNN streams are used to deal with different view image sequences. For each view image sequence captured from a camera, a 3D-CNN is used to extract the spatial-temporal information for the moving object.
- A novel view-pooling method named weighted average view-pooling (WAVP) is designed to fuse the multiple view information and learn view-related features.
- The MV3D-CNN model is evaluated using the ASLLVD corpus for sign language recognition and applied to a real-world moving vehicle recognition task. The experimental results indicate that the model offers significant improvements over baseline methods.

The remainder of this paper is structured as follows. Section 2 provides a thorough literature review on the details of 3D-CNNs. Section 3 describes our proposed MV3D-CNN model. In Sect. 4, the results of two multi-view MOR experiments are presented, enabling us to evaluate the model performance. Section 5 draws together some conclusions from this study.

2 Preliminaries

2.1 Notation

- The training dataset consists of multi-view videos captured from different cameras. For a dataset S, the mth sample in S is denoted as $\{(x^{(1)}(m), x^{(2)}(m), ..., x^{(v)}(m)), d(m)\}$, where $x^{(v)}(m)$ is the input of view v, and $d(m)$ is the corresponding label.
- We use x_{ijk} to denote the input to the 3D convolution layer, and z_{ijk} to denote the input to the 3D pooling layer at position (i, j, k). z_{ijk} is also the output from the 3D convolution layer.
- g denotes the output of the 3D pooling layer. a is the output from the view-pooling layer and the input to a fully connected neural network (FNN).
- y is the output from the softmax classifier.
- w^{opq} denotes the weight at position (o, p, q) of the kernel, and b is the bias parameter. β is the mutiplicative bias of the pooling layer. θ represents the parameters of the softmax layer, and α indexes the weights of the view-pooling layer.
- $f(\cdot)$ denotes the activation function, which is the *tanh* function in the 3D convolution layer and *ReLU* function in the FNN. $down(\cdot)$ represents a subsampling function. Typically, this function aggregates values over each distinct n-by-n-by-n block in the input image, so that the output of this operation is n-times smaller in both the temporal and spatial dimensions.
- O, P, Q denotes the size of the 3D kernel along the temporal dimension and the image's width and height, respectively.
- λ is the weight decay parameter, $1\{\cdot\}$ is the characteristic function, and l is the index of lth layer.

2.2 3D convolutional neural networks

Before introducing the proposed model, we present a short overview of 3D-CNNs. The 3D-CNN model is a variant of CNNs developed for sequence learning tasks. 3D-CNNs have been broadly applied in video classification and motion recognition tasks [3, 10]. Most 3D-CNNs treat the data sequence as a fixed-frame sequence and apply 3D

convolution layers for feature learning. 3D convolution kernels can learn more complicated spatial and temporal features together. The corresponding 3D pooling neural layers are utilized to reduce the number of parameters in the spatiotemporal lever. 3D-CNNs usually have two different layers: the 3D convolution layer and 3D pooling layer. The function of the 3D convolution layer is to apply several convolutional filters over the input volumes. The 3D pooling layer selects the best feature extracted by the 3D convolution layer.

2.2.1 The 3D convolution layer

In a 3D convolutional neural layer, there are many 3D convolution kernels, each of which is replicated over the image frame sequence. The 3D convolution layers play an important role in extracting spatiotemporal features, because they can encode temporal visual information from the video clips. The 3D convolution layer can be computed as follows:

$$z_{ijk} = f\left(\sum_{o=0}^{O-1}\sum_{p=0}^{P-1}\sum_{q=0}^{Q-1} w^{opq} \cdot x_{(i+o)(j+p)(k+q)} + b\right). \quad (1)$$

2.2.2 The pooling layer

To reduce the feature dimension, the max or average value of a special feature is calculated by pooling layers over all the pixels of the image. Pooling methods include *Max pooling* and *Average pooling*. The pooling layers are useful in reducing the number of parameters in the network by reducing the spatial size of the vector representation. 3D pooling layers are usually connected to 3D convolution layers. The 3D pooling layer is computed as:

$$g = f(\beta \cdot down(z) + b). \quad (2)$$

In deep 3D-CNNs, the 3D pooling layer is usually applied after the 3D convolution layer. Thus, in this paper, the output of the 3D convolution layer is considered the input of the 3D pooling layer.

3 Multi-view 3D convolutional neural networks

The focus of this paper is exploiting the ability of 3D-CNNs and combining multi-view learning for MOR tasks. Refer to [13], the basic approach for multi-view learning is to distribute all visual views into several single-view descriptors, and then concatenate those descriptors to obtain final recognition results of the learning tasks. The success of multi-view learning approaches mostly depends on the *consensus* and *complementary* principles. The first principle aims to balance the view information of different views, whereas the latter ensures that complementary information inside multi-view data can be exploited to comprehensively and accurately extract features and thus improve the learning performance. Further details of multi-view learning can be found in [15]. The basic solution of multi-view learning is to use an independent feature extractor to model a particular view and to optimize these functions to exploit the view-related information of input data to improve the learning performance. The multiple views for different problems have different formats, e.g., color descriptor, local shape descriptor, and spatiotemporal context captured by multiple cameras. In this paper, we use 3D-CNNs for MOR tasks as the basic unit of a local shape descriptor.

A naive method to combine 3D-CNNs and multi-view learning is to train single 3D-CNNs for each view and then average all of the 3D-CNNs from all views as the final result. Though this method achieves the basic requirement of using multiple views, it does not incorporate the principles of multi-view learning. Because all 3D-CNNs are trained on different views independently, this method cannot assist in the learning of view-related features. Facing this challenge, we consider how to use a parallel 3D-CNN model to combine the view descriptors from all separate 3D-CNNs to give a union architecture. The simplest method is to directly concatenate all view descriptors and then connect them to a unified classifier for the final recognition task. The training of this unified model could help in finding view information.

Though parallel architectures using separate descriptors would possibly assist with the MOR task, a considerable problem is that simply concatenating the multiple view descriptors may not benefit the final recognition. As each individual view with incomplete information participates in recognition, this method could therefore easily confuse classes if some views were irrelevant to the task. To deal with this challenge, a view-pooling layer has been developed to aggregate the global view information [14]. The basic idea is to use an element-wise maximum operation or average operation across the views in the view-pooling layer. The view-pooling layers are closely related to max pooling and maxout layers, which would also be applied in the 3D pooling layers.

The details of our architecture are depicted in Fig. 1. The model contains three important parts. First, the MV3D-CNN model involves independent 3D-CNNs to extract features from videos taken at multiple views. Second, a view-pooling layer is developed to aggregate all view descriptors and learn view-related features. Finally, an FNN with a softmax classifier performs the final recognition. This type of deep architecture has demonstrated strong performance in many vision tasks [8].

The view-pooling layer is intended to enable the model to distinguish important views from irrelevant views. Thus,

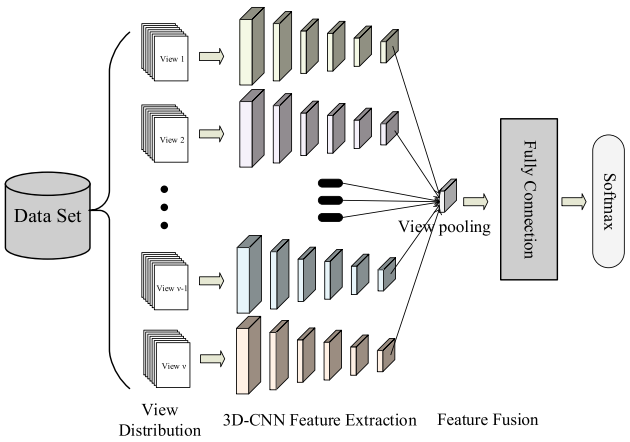

Fig. 1 Architecture of multi-view 3D convolutional neural networks

the model can choose several of the most important views as the final determinant. We design a new variation of the average view-pooling layer, called the weighted average view-pooling (WAVP) layer, to help aggregate the view descriptors. The output from all multiple view descriptors is usually the output of the last 3D convolution layer. The three view-pooling layers are computed as:

MVP layer:

$$a_{ijk} = \max\{z^1_{ijk}, z^2_{ijk}, ..., z^v_{ijk}\}. \tag{3}$$

AVP layer:

$$a_{ijk} = \frac{1}{v}\sum_u z^u_{ijk}. \tag{4}$$

WAVP layer:

$$a_{ijk} = \frac{\sum_t \exp^{\alpha^t} \cdot z^t_{ijk}}{\sum_u \exp^{\alpha^u}}. \tag{5}$$

Different from other view-pooling methods, WAVP adds the weighted parameters to confirm that the network has automatically extracted view-related features when aggregating the multiple view descriptors. Training the WAVP layer using a gradient descent method ensures that important view features stand out. When the multiple views are reduced to two views, the parameters of the WAVP layer can be set to α and $1-\alpha$, $\alpha \in (0,1)$. The 3D pooling layer can be computed as::

$$a_{ijk} = \alpha \cdot x^{(1)}_{ijk} + (1-\alpha)\cdot x^{(2)}_{ijk}. \tag{6}$$

After fusing the view-pooling layer, the aggregate of the multiple views is input to the FNN for further deep learning of high-level features. The formulation at layer l of the FNN is as follows:

$$a^l = f\left(w^l a^{l-1} + b^l\right). \tag{7}$$

We denote the output of the final layer of the FNN as a^L. The cost function can be defined using the softmax classifier as:

$$J(\theta) = -\frac{1}{m}\sum_m\sum_i 1\{d(m)=i\}log\frac{\exp^{\theta_i a^L(m)}}{\sum_j \exp^{\theta_j a^L(m)}} + \frac{\lambda}{2}\sum_i\sum_k \theta^{(2)}_{ik}. \tag{8}$$

By maximizing the cost function, the MV3D-CNN model learns how to extract spatiotemporal features and fuse the view-related features into a deep FNN structure for the final recognition. The steps in the MV3D-CNN model for MOR tasks are formulated in Algorithm 1. The training of the network is based on the variant of backpropagation algorithm, for details refer to [16]. Although our proposed architecture is similar to previous parallel architectures [4, 10, 14], there are three significant and interesting improvements. First, we use 3D-CNNs for the single-view video feature descriptor to learn spatiotemporal features, but this only involves parallel architectures at the image level to solve MOR problems. Second, we use a deep FNN architecture to improve learning performance, as more high-level features could be learned to enhance performance. Finally, we propose a new view-pooling layer for combining the multiple view information. The proposed view-pooling layer learns to select important view information for later FNNs to assist with view-related feature learning.

Algorithm 1 The training algorithm of MV3D-CNNs with a WAVP layer

Input The training dataset S.
Output The network's parameters.
Initialization $w \leftarrow w_0, b \leftarrow b_0, \alpha \leftarrow \alpha_0, \beta \leftarrow \beta_0, \theta \leftarrow \theta_0$.
for $epoch = 1$ to max_epoch **do**
  Define the cost function $J(\theta)$ by equation 8.
  Forward pass.
  for $l = 1$ to max_l in single-view feature learning stage **do**
    for $v = 1$ to max_v **do**
      if l is the convolution layer **then**
        Compute the activations of convolution layer by equation 1.
      else
        Compute the activations of pooling layer by equation 2.
      end if
    end for
  end for
  View Aggregation.
  Aggregate the multiple view information by equation 5.
  Compute the activation of FNN layers.
  Backward pass
  Use the variational backpropagation algorithm to compute the partial derivatives of
  parameters $w, b, \alpha, \beta, \theta$.
  Update Weights.
  Update all parameters by gradient descent method.
end for

4 Experiments

In this section, we describe the application of MV3D-CNNs in two experiments. First, we examine the performance of the proposed method on the American Sign Language Lexicon Video Dataset (ASLLVD) corpus [17, 18] and compare the results with those from other baseline methods. Second, we apply the MV3D-CNNs to a complicated real-world vehicle recognition problem.

4.1 Multi-GPU computation

The experiments were implemented using the Torch toolbox.[1] We designed the MV3D-CNN architecture and implemented the WAVP layer for this architecture. The model was trained on multiple GPUs for two MOR tasks. The average training time for the ASLLVD corpus was approximately 10 h, and that for the vehicle recognition task was 2 h.

4.2 Baselines

To compare the MV3D-CNNs with general CNN methods, we report the accuracy of the single-view CNNs and the single-view 3D-CNN approaches on the ASLLVD corpus. The single-view methods use data from one view to extract features and then apply the FNN and softmax classifier for the final recognition. The CNNs were trained using a single frame of the video clips. The trained CNNs were tested on all frames of each video sample, and the results were averaged to give the final scores. Simple concatenation and three view-pooling methods were also applied in the sign language recognition task. In the real-world moving vehicle recognition task, we applied the proposed MV3D-CNN with WAVP using different FNNs. The experimental results for MV3D-CNN with different 3D-CNN layers are also reported for this task. ReLU and drop out techniques [19] inside the FNNs were used to decrease the training time and prevent over-fitting. Model parameters such as the learning rate, weight decay, and the number of neurons in all layers were tuned to their optimal values for comparison. The best classification results are highlighted with bold face in Tables 1 and 3.

4.3 Sign language recognition

The ASLLVD corpus consists of videos of more than 3300 ASL signs. There are a total of almost 9800 samples performed by 6 native ASL signers. The data were collected at Boston University. This dataset contains several synchronized videos of the signs from different angles that adapt to our architecture. The videos were captured from four synchronized cameras, affording a full-resolution front view, a side view, a close-up of the head region and a half-speed high-resolution front view of the signer.

4.3.1 Datasets

In applying our proposed model, there are two major problems with the ASLLVD corpus. First, the corpus is deficient in samples for each sign, because there are only 6 native ASL signers and each sign has an average of three samples. Second, the major sign motion focuses on the hand shapes of the signers. It is necessary to capture hand shape fragments from the source corpus. Considering the proposed problems, we applied a preprocessing stage to the datasets. Primarily, we selected 50 sign language labels that had been applied to more than five samples in the ASLLVD corpus. We created video clips with an average of 150 frames for each sample. We then chose data samples by drawing a box including the hand shape inside the image frame. Specifically, we invited 10 volunteers to capture image fragments from the source images. Some image clips of the sign for *afraid* are described in Fig. 2. Each volunteer captured hand shape image fragments with eight different capture rules. The capture rules related to the box size (e.g., fixed-box and adaptive box), body parts (e.g., hand and shoulder), and body areas (e.g., left body part, right body part, or both). Details of the capture rules are depicted in Fig. 3. Thus, we obtained over 400 training samples for each sign label.

4.3.2 Experiment setup

Before building the dataset, we examined the captured samples and removed poor or wrongly captured samples. We then designed two experiments for the ASLLVD corpus, 10-label and 50-label SLR. The 10-label SLR contains 4500 two-view videos (150 frames/video on average), annotated into 10 sign classes. The 50-label SLR includes 22,300 two-view videos of 50 signs. Among the video clips, the hand shapes of several start frames and end frames contain almost no motion or action. Thus, we selected the middle 64 frames as the temporal sequence. The task was to recognize the sign from the two-view image sequences. A sign language sample named *abuse* captured with the rule *right hand fixed-box* is depicted in Fig. 4.

4.3.3 Quantitative analysis

The best training parameters are presented in Fig. 5. We used three 3D-CNN convolution layers and two 3D-CNN pooling layers as the separated view feature learning descriptors and then applied a WAVP layer to aggregate

[1] http://torch.ch/. Torch is a scientific computing framework, which supports deep learning models and algorithms.

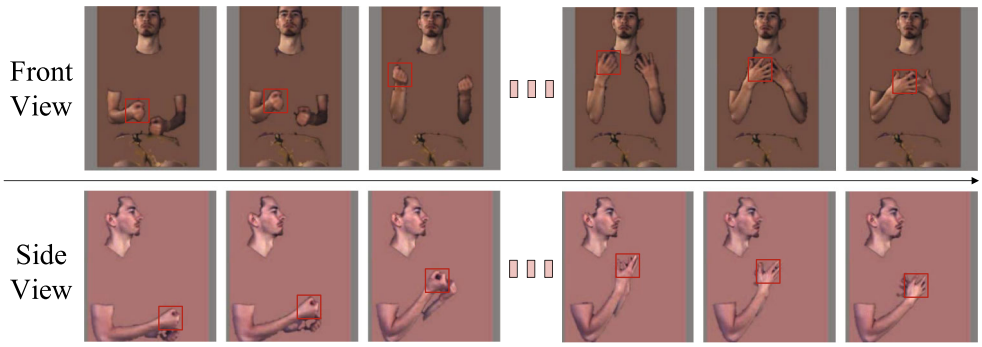

Fig. 2 Sign *afraid* of multi-view image sequence

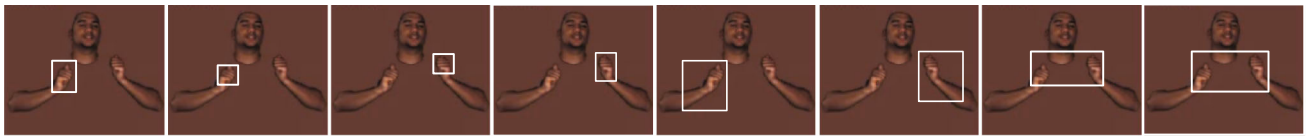

Fig. 3 Capture rules have 8 categories including: *right hand suitable-box*; *right hand fixed-box*; *left hand suitable-box*; *left hand fixed-box*; *right shoulder suitable-box*; *left shoulder suitable-box*; *double hands fixed-box*; and *double hands suitable-box*

Fig. 4 Sample of sign *"abuse"*

the two views. This allowed us to construct the inner view information and extract view-related features. A four-layer (for 10-label) or five-layer (for 50-label) FNN with a softmax classifier was connected to the view-pooling layer for the final recognition. All other baseline methods used a similar number of neuron connections. The experimental results are presented in Table 1. Our MV3D-CNN exhibits superior performance to that of the four nave baselines. Compared with other view-pooling methods, the experimental results suggest that the WAVP layer gives the best performance in the MV3D-CNN model. It is clear that the recognition accuracy from the front view is higher than that for the side view. It is understandable that the front view contains more useful information than the side view in the ASLLVD corpus.

4.4 Vehicle recognition

Vehicle recognition is an important and challenging problem entailing the use of video data collected from highways and railroads. This problem can be formulated as the multi-class classification of datasets involving dynamic moving objects. The recognition procedure is quite challenging because of the complex highway environment and varying vehicle appearance within categories. The source data and application requirements are supported by the Highway Management Department (HMD) of Jiangxi Province in China. The source data include 10-day surveillance video data, and the task is to recognize moving vehicles from these data. Specifically, the task is to recognize the vehicle categories stipulated by the highway toll criterion of HMD.

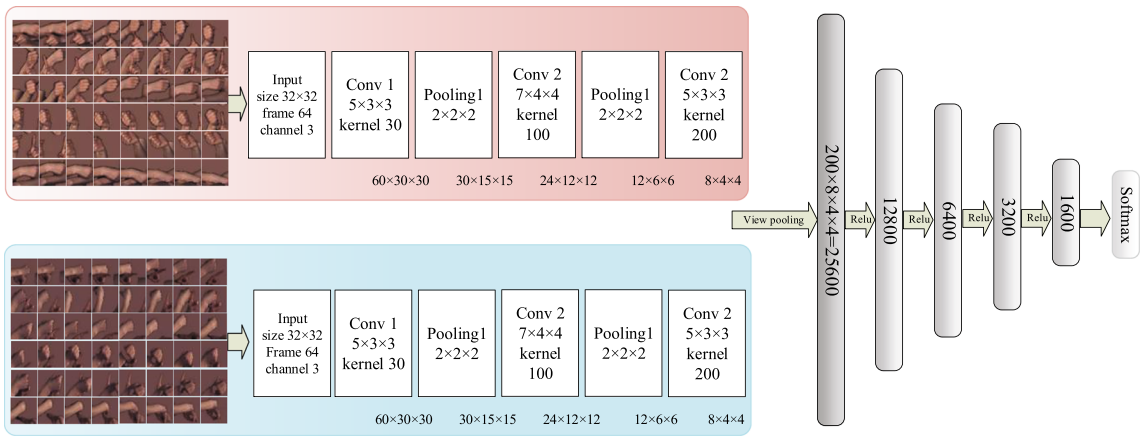

Fig. 5 Best network parameter settings in sign language recognition experiment

Table 1 Results of ASLLVD corpus for sign language recognition

Model	10-L accuracy (%)	50-L accuracy (%)
Side-view average CNNs	81.57	78.03
Front-view average CNNs	83.85	80.81
Side-view 3D-CNNs	86.92	82.28
Front-view 3D-CNNs	87.45	84.21
MV3D-CNNs with simple concatenation	88.64	85.49
MV3D-CNNs with a MVP layer	92.93	90.10
MV3D-CNNs with a AVP layer	89.44	88.74
MV3D-CNNs with a WAVP layer	**93.52**	**91.58**

There are a total of four types of coaches and five types of trucks (see Table 2 for more details).

4.4.1 Datasets and training

The video data were captured from cameras at different locations. We selected front-view and back-view daytime videos and used these to create video clips. Considering the resolution of the images, gray images were captured from the front-view videos and color ones were captured from the back-view videos. The captured video clips were resized to 32 × 32 pixels. Similar to the SLR task, we extracted the middle 32 frames as the temporal sequences for building the datasets. There are a total of 9000 data samples in our experiments: 80 % of these were used for training, 5 % were used for variation, and the remainder were used for testing.

Table 2 Coaches are categorized according to their passenger capacities (C), and trucks are categorized according to their weight of goods capacities (W, the unit = 1 ton)

Categories	Passenger capacities
Coach 1	$C \leq 7$
Coach 2	$7 < C \leq 19$
Coach 3	$19 < C \leq 39$
Coach 4	$C > 39$
Categories	**Weight capacities**
Truck 1	$W \leq 14$
Truck 2	$14 < W \leq 17$
Truck 3	$17 < W \leq 25$
Truck 4	$25 < W \leq 32$
Truck 5	$W > 32$

4.4.2 Quantitative analysis

This experiment compared MV3D-CNNs with different deep layers. The best results are shown in Fig. 6. The experimental results show that the model can achieve accuracy of up to 95.37 %. As the model became deeper, the accuracy did not improve. This is mainly because of the scale of the datasets (Table 3).

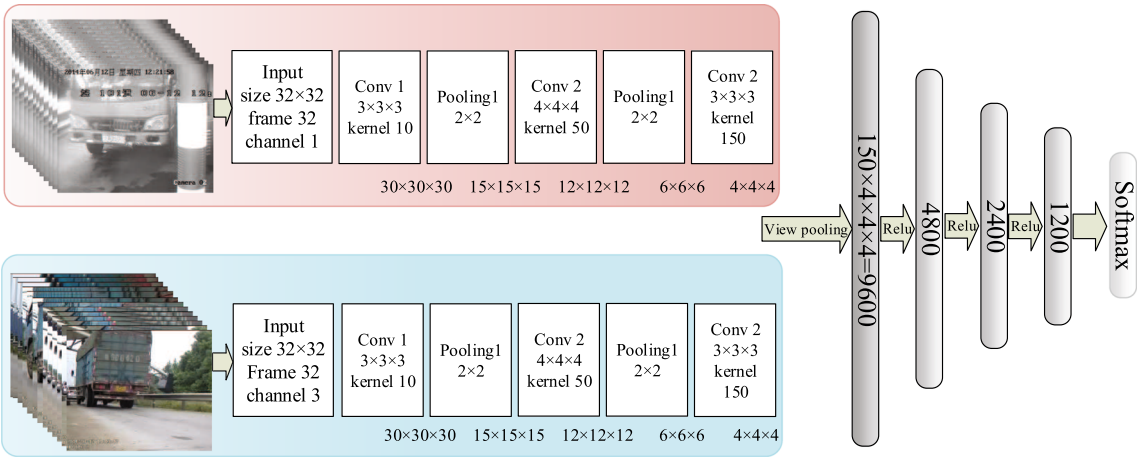

Fig. 6 Architecture of best result in moving vehicle recognition

Table 3 Experiment results in the vehicle recognition task by MV3D-CNNs with different network layers

Model	Accuracy (%)
MV3D-CNNs (4-layer CNNs + 4-layer FNNs)	91.38
MV3D-CNNs (4-layer CNNs + 6-layer FNNs)	92.21
MV3D-CNNs (5-layer CNNs + 4-layer FNNs)	94.98
MV3D-CNNs (5-layer CNNs + 5-layer FNNs)	**95.37**
MV3D-CNNs (5-layer CNNs + 6-layer FNNs)	95.02

5 Conclusion

This paper has described an MV3D-CNN for MOR tasks. The model constructs features from multiple views and uses 3D-CNNs to extract spatiotemporal features. We designed a novel view-pooling method to assist the learning of view-related features. The model was evaluated using sign language recognition and moving vehicle recognition tasks. The experimental results indicate that the model outperforms conventional methods on the ASLLVD dataset. In the vehicle recognition task, the model produced superior performance in a real-world application environment. As multi-view learning requires datasets that have multiple views, we have only evaluated the model on two small-scale datasets containing two views. It would be interesting to extend the model to large-scale MOR tasks. The learning stage was conducted without pre-training using some unsupervised algorithms. We will apply the MV3D-CNN model to large-scale datasets in future work.

Acknowledgments This work was supported by the National Science Foundation of China (Grant Nos. 61432012 and 61402306).

References

1. Kuehne H, Jhuang H, Garrote E, Poggio T, Serre T (2011) A large video database for human motion recognition. In International Conference on Computer Vision
2. Yang J, Yuan J, Li YF (2015) Flexible trajectory indexing for 3d motion recognition. In IEEE Winter Conference on Applications of Computer Vision, pp 326–332
3. Ji S, Xu W, Yang M, Yu K (2013) 3d convolutional neural networks for human action recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp 221–231
4. Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. In Advances in Neural Information Processing Systems, pp 568–576
5. Jang S, Choi K, Toh K, Teoh ABJ, Kim J (2015) Object tracking based on an online learning network with total error rate minimization. Pattern Recognition, pp 126–139
6. Baccouche M, Mamalet F, Wolf C, Garcia C, Baskurt A (2010) Action classification in soccer videos with long short-term memory recurrent neural networks. In Artificial Neural Networks-ICANN, pp 154–159
7. Ciresan DC, Meier U, Masci J, Maria Gambardella L, Schmidhuber J (2011) Flexible, high performance convolutional neural networks for image classification. In IJCAI Proceedings-International Joint Conference on Artificial Intelligence, p 1237
8. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pp 1097–1105
9. LeCun Y, Kavukcuoglu K, Farabet C (2010) Convolutional networks and applications in vision. In Proceedings of 2010 IEEE International Symposium on Circuits and Systems, pp 253–256
10. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In IEEE Conference on Computer Vision and Pattern Recognition, pp 1725–1732
11. Yue-Hei Ng J, Hausknecht H, Vijayanarasimhan S, Vinyals O, Monga R, Toderici G (2015) Beyond short snippets: Deep networks for video classification. arXiv preprint arXiv:1503.08909
12. Tran D, Bourdev L, Fergus R, Torresani L, Paluri M (2015) Learning spatio-temporal features with 3D convolutional networks
13. Chang X, Dacheng T, Chao X (2013) A survey on multi-view learning. arXiv preprint arXiv:1304.5634

14. Su H, Maji S, Kalogerakis E, Learned-Miller E (2015) Multi-view convolutional neural networks for 3D shape recognition. arXiv preprint arXiv:1505.00880
15. Zhe W, Songcan C, Daqi G (2011) A novel multi-view learning developed from single-view patterns. Pattern Recognition, pp 2395–2413
16. Bouvrie J (2006) Notes on convolutional neural networks
17. Neidle C, Vogler C (2012) A new web interface to facilitate access to corpora: Development of the asllrp data access interface. In Proceedings of the International Conference on Language Resources and Evaluation
18. Neidle C, Thangali A, Sclaroff S (2012) Challenges in development of the american sign language lexicon video dataset (asllvd) corpus. In Processings of Sign Languages: Interactions between Corpus and Lexicon
19. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov R (2012) Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580