Continuous Chinese sign language recognition with CNN-LSTM

Su Yang, Qing Zhu

Su Yang, Qing Zhu, "Continuous Chinese sign language recognition with CNN-LSTM," Proc. SPIE 10420, Ninth International Conference on Digital Image Processing (ICDIP 2017), 104200F (21 July 2017); doi: 10.1117/12.2281671

Event: Ninth International Conference on Digital Image Processing (ICDIP 2017), 2017, Hong Kong, China

Continuous Chinese Sign Language Recognition with CNN-LSTM

Su Yang, Qing Zhu
Faculty of Information Technology, Beijing University of Technology, No. 100 Ping Leyuan, Chaoyang Disctrict, Beijing 100124, China

ABSTRACT

The goal of sign language recognition (SLR) is to translate the sign language into text, and provide a convenient tool for the communication between the deaf-mute and the ordinary. In this paper, we formulate an appropriate model based on convolutional neural network (CNN) combined with Long Short-Term Memory (LSTM) network, in order to accomplish the continuous recognition work. With the strong ability of CNN, the information of pictures captured from Chinese sign language (CSL) videos can be learned and transformed into vector. Since the video can be regarded as an ordered sequence of frames, LSTM model is employed to connect with the fully-connected layer of CNN. As a recurrent neural network (RNN), it is suitable for sequence learning tasks with the capability of recognizing patterns defined by temporal distance. Compared with traditional RNN, LSTM has performed better on storing and accessing information. We evaluate this method on our self-built dataset including 40 daily vocabularies. The experimental results show that the recognition method with CNN-LSTM can achieve a high recognition rate with small training sets, which will meet the needs of real-time SLR system.

Keywords: Sign language recognition, convolutional neural network, recurrent neural network, Long Short-Term Memory

1. INTRODUCTION

Sign language, as an necessary communication tool for deaf-mute people, is composed of complicated hand shape, movement, location of hand, facial expression, etc. It's difficult for normal people to understand without requisite knowledge. Sign language recognition (SLR) aims at translating sign language into text or speech, which can provide a convenient platform for daily conversations between the normal and the deaf-mute. Furthermore, with the development of artificial intelligence, human-computer interaction attracts much more attention than before. And SLR plays a critical role in this area nowadays.

The research began in 1990's. 1988, Tamura et al. assumed that sign language consisted of hand shape, movement, and location of the hand. They extracted 3-D feature of these factors, converted it into 2-D image features, and classified the motion image of sign language with the 2-D features[1]. A recognition method based on Hidden Markov Model (HMM) was proposed in this field by Starner et al. in 1995. For capturing hand shape, orientation and trajectory, they used colored gloves as an assistant tool. This method performed well in experiments on 40 vocabulary words[2]. And HMM has been employed in SLR increasingly since then. With the assistant of various features and techniques, HMM showed effectiveness in continuous and real-time SLR tasks [3, 4, 5]. Keskin et al. created a realistic 3D hand model to represent hand and trained random decision forests (RDF) on depth images. With RDF, they implemented pixel classification and exploited the results to estimate the joint locations for the hand skeleton. A support vector machine (SVM) based module was described for recognition and achieved high accuracy in experiments on 20000 images of American Sign Language digits [6]. Mekala et al. applied combinational neural network (NN) to recognize alphabets of sign language. The advantage of this algorithm was high processing speed which can satisfied the requirement of real-time system[7]. Sawant et al. built a real-time system using Principle Component Analysis (PCA) algorithm to convert gestures into text and voice. With Eigen values and Eigen vectors as features, this system was capable of recognizing 26 gestures from the Indian Sign Language[8].

For the Chinese Sign language (CSL) recognition, many studies have been involved in the field. Gao et al. combined Artificial Neural Networks (ANNs) and Dynamic Programming (DP) to code a sign. With HMM, this approach was effective in recognizing large vocabulary CSL[9]. Zhang et al. designed a framework applied decision tree and multistream HMM to recognizing the information fusion of sensors. And they built a real-time interactive system that convert hand gestures into control commands[10]. Zhang et al. proposed a fast recognition method for CSL letter spelling alphabet and

Ninth International Conference on Digital Image Processing (ICDIP 2017), edited by Charles M. Falco, Xudong Jiang
Proc. of SPIE Vol. 10420, 104200F · © 2017 SPIE · CCC code: 0277-786X/17/$18 · doi: 10.1117/12.2281671

digits utilizing relief algorithm. This algorithm segmented the hand region of given gesture images with complexion model[11].

In this paper, we built a CSL recognition approach based on convolutional neural network (CNN) and Long Short-Term Memory (LSTM) network. Unlike the methods mentioned above, CNN uses images as input directly with robustness, which can avoid the complex hand gesture segmentation and feature extraction. It can exploit the information as much as possible. Since there are many dynamic hand gestures in CRL, a continuous recognition method is more practical for the recognition task. As a recurrent neural network (RNN), LSTM network can use the contextual information of sequences during learning and is very suitable for time series problems. And it breaks the limit of traditional RNN for accessing the context in practice, with tackling the vanishing gradient problem[12]. We utilize CNN to generate vectors representing the images captured from CSL video, and feed them into LSTM model in sequence for recognition. To reduce the amount of computation during recognition, we localize the hand of the subject in video, and capture the images centered on hand location sequentially. We also preprocess the images, in order to facilitate the information learning of the CNN-LSTM model.

2. CNN-LSTM MODEL

2.1 Architecture Overview

The CNN-LSTM model framework employed in our recognition method is as shown in Figure 1. We use CNN to transform the images after preprocessing into the vector representations. The vectors are the input of the LSTM network. CNN architecture is trained in advance, with the labeled images captured from CSL video. In this model, we only use the convolutional layers and max-pooling layers before the flatten layer, and the outputs of flatten layer are the vectorize representations of images. For a CSL vocabulary video, there is a set of vectors that feed into LSTM sequentially. LSTM network uses the input vectors and context information to recognize the sign language gesture. Each hidden state of the LSTM emits a prediction. It can help to recognize which word the next movement belongs to. Since there are a large number of similar movements in gestures of CSL, LSTM can improve the recognition accuracy with continuous input images.

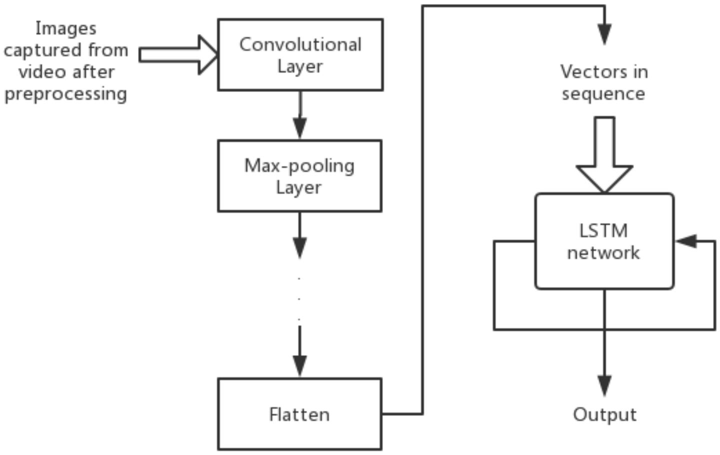

Figure 1. CNN-LSTM model framework

2.2 Convolutional Neural Network

CNN is a special multilayer neural network designed to process two dimensional data. With the basic architecture idea, sparse connectivity and shared weights, the training parameters of network can be reduced, which make CNN has strong adaptability. And CNN can learn appropriate features from training samples itself, so the traditional feature extraction can be avoided[13]. The advantage of using CNN in SLR is that we can avert the complex hand segmentation and manual feature extraction, which can reduce the information loss and improve the accuracy of recognition.

CNN is composed of convolutional layers, max-pooling layers and fully connected layers. The convolutional layers execute convolution on inputs with kernel, and get the feature maps of previous layer. The convolution function is defined as (1).

$$a_j^l = f(\sum_{i \in M_j} a_i^{l-1} * k_{ij}^l + b_j^l) \tag{1}$$

where $*$ is the convolution operation, *l* is the layer number, *j* is the number of output map, a_j^l is the output map, M_j is the set of input maps, k_{ij}^l is the kernel of *l*-th layer in the network, b_j^l is the bias of output, and *f* is activation function.

The max-pooling layer is a form of subsampling layer. The subsampling operation divides the input feature map into a set of regions and calculates the max or mean value over the region. The values of the sub-region make up the output map of subsampling layer. The function of subsampling layer can be described as follow:

$$a_j^l = g(a_i^{(l-1)}), \forall i \in R_j \tag{2}$$

where g is the subsampling operation and R_j is the subsampling region.

The framework used in our work is shown in Table 1, 2. The two frameworks are for producing the vectors with size 1024 and 128 respectively. The four columns in tables indicate layer type, kernel size, output size and parameter amount of layer. For the framework in Table 1, the size of input images is 256×256×1. The first convolutional layer has 16 kernels of size 5×5. The patch size of first max-pooling is 5×5. The second convolutional layer filters the input map with 32 kernels of size 5×5, and the third one uses 64 kernels of size 3×3. These two convolutional layer are followed by max-pooling layer with size 3×3. A rectification activation function (RELU) is used in the hidden weight layers following convolutional layers[14]. The dropout rates in the architecture is set to 0.2 and 0.5 for the flatten layer and full connected layer. And the loss function used in output layer is log-likelihood function, which corresponds to the softmax function. ADADELTA [15] are chosen as the learning algorithm. The framework in Table 2 add a fourth convolutional layer with 128 kernels of size 3×3 on the framework in Table 1, which is followed by a max-pooling with patch size 2×2.

CNN will be trained in advance and the parameters will be saved. In our recognition method, we only employ the layers above the flatten layer (including the flatten layer). The output feature map size of flatten layer are 1024 and 128 respectively, corresponding to different vector size. With the pre-training parameters, CNN extracts the feature map of images and transforms them into vectors, which is the input of LSTM network.

Table 1.CNN framework for output vector of size 1024.

Layer type	Kernel size	Output size	Param #
input	-	256×256×1	-
convolution	16@5×5	16@252×252	0.4K
max-pooling	16@5×5	16@50×50	0.4K
convolution	32@5×5	32@46×46	0.8K
max-pooling	32@3×3	32@15×15	0.2K
convolution	64@3×3	64@13×13	0.5K
max-pooling	64@3×3	64@4×4	0.5K
flatten	-	1024	-
dropout(0.2)	-	1024	-
fullyconnected	-	64	65K
dropout(0.5)	-	64	-
softmax	-	40	2.6k

Table 2.CNN framework for output vector of size 128

Layer type	Kernel size	Output size	Param #
input	-	256×256×1	-
convolution	16@5×5	16@252×252	0.4K
max-pooling	16@5×5	16@50×50	0.4K
convolution	32@5×5	32@46×46	0.8K
max-pooling	32@3×3	32@15×15	0.2K
convolution	64@3×3	64@13×13	0.5K
max-pooling	64@3×3	64@4×4	0.5K
convolution	128@3×3	128@2×2	1.1K
max-pooling	128@2×2	128@1×1	0.5K
flatten	-	128	-
dropout(0.2)	-	128	-
fullyconnected	-	64	8K
dropout(0.5)	-	64	-
softmax	-	40	2.6k

2.3 LSTM network

LSTM architecture is a variant of RNN, which is designed to confront the vanishing gradient problem. It uses memory blocks to store and access information over long periods of time. So LSTM is very suitable for continuous recognition tasks, which demand for using long-term contextual information[12].

The formulas to update LSTM at time *t* are described as follow,

$$i_t = \sigma_i(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1}) \tag{3}$$

$$o_t = \sigma_o(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1}) \tag{4}$$

$$f_t = \sigma_f(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1}) \tag{5}$$

$$c_t = f_t c_{t-1} + i_t \sigma_c(W_{xc}x_t + W_{hc}h_{t-1}) \tag{6}$$

$$h_t = o_t \sigma_t(c_t) \tag{7}$$

where σ is the non-linear function, W is the weight between two connected units, x_t is the input vector, and i_t, o_t, f_t, c_t, h_t represent outputs of input gate, output gate, forget gate, cell and hidden state vector respectively.

In our model, the weights are initialized with uniform random numbers of scale 0.01. The $\tanh(\cdot)$ activation function is used in hidden layer. To avoid the network overfitting, the dropout is employed between the input vector and the input gate, with dropout rate 0.5. Softmax function is used to predict the distribution with the output of previous layers, and the likelihood loss function is the cost function of our model. To train the LSTM model, RMSProp technique[16], which is the optimization of gradient descent, is applied to minimize the loss function. The back propagation algorithm is used to update the weights.

3. PREPROCESSING

As a continuous SLR approach, our system take CSL videos as input directly. So we must capture the frames and preprocess them, in order to provide materials for the CNN model. The steps of preprocessing is as shown in Figure 2. Considering the change frequency of movement in CSL, the frame rate for capture is 10fps.

During the phase of hand detection, we use the skin-color detection algorithm to detect the hand region, which has been widely used in hand location and gesture recognition[17]. Since the facial area may interfere the hand detection, a convenient cascade classifier provided by Opencv is utilized to remove this area[18]. After the skin-color detection, we calculate the close contours of each skin color areas. Since the image may have one or two hands, we use the two largest contours to represent the hand region. The reason for this is that the facial area has been removed, so the hand region is the largest skin-color area. We calculate up-right bounding rectangle of the region, and the center of rectangle is the center of hand region, which will be used to extract the upper body image of subject.

Since the static background may cause CNN model overfitting and lower the recognition rate, it's necessary to remove the image background. We calculate the mean upper body image over our datasets beforehand, and subtract the mean from the extraction image. The result of subtraction could be negative numbers, which cannot be stored as pixels. Since the absolute values of them are very small in practice, these numbers are saved as 0.

In the last stage, we transform the images into the Hue-Saturation-Value (HSV) color space, which is closely to the human perception of color[19]. To reduce the training cost of following networks, we just choose hue component to build the single channel images as the CNN input.

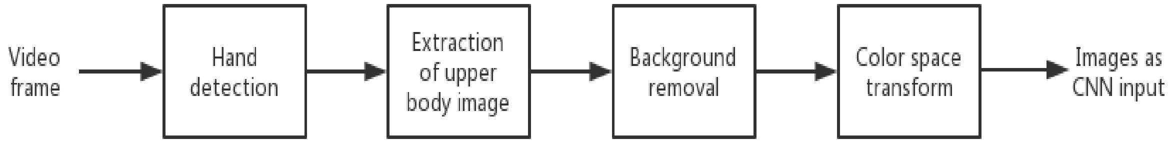

Figure 2. Preprocessing flow

4. EXPERIMENTAL RESULTS

4.1 Datasets

Experiments in our paper are performed on a Chinese sign language instructional video material named *We Learn Sign Language*. It's a popular instructional materials in China and can be free download online. We choose 40 daily vocabularies in Chinese sign language, and obtain 320 corresponding gesture videos to build our dataset, 256 for training and 64 for test. These videos are more than 1s length. The training data of CNN are 4000 labeled images captured from videos.

4.2 Results

The experimental results are shown in Figure 3 and Table 3. Figure 3 illustrates the loss decreasing and accuracy increasing with different input vector sizes. For the vector size 128, the loss values smoothly decline and accuracies increase gradually during training. In contrast, there are a few shaking in the loss and accuracy curves with vector of size 1024. And the validate accuracy increases much faster for vector size 128, which is over 90% in 6th epoch. The training and test accuracies with two vector size are described in Table 3. All accuracies reach 95%, and apparently the recognition with vector of size 128 is more effective. So the CNN model with four convolutional layers is more appropriate for the feature extraction in our method. And the recognition method we built is quite efficient for continuous CSL recognition task.

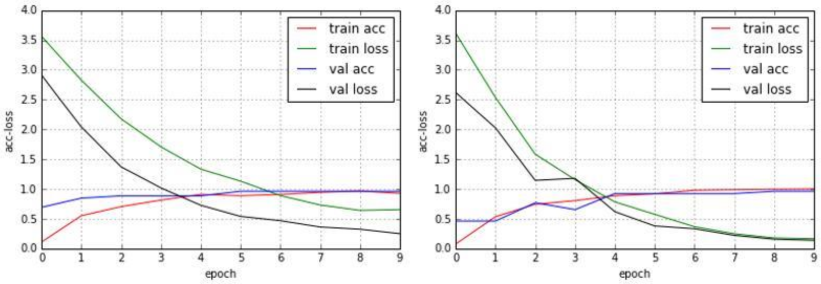

Figure 3. Lost decreasing and accuracy increasing of training set and validate set with vector size 128 and 1024

Table 3.Training and test accuracy.

	Training accuracy	**Test accuracy**
Input vector of size 128(%)	96.52	100
Input vector of size 1024 (%)	95.14	98.43

5. CONCLUSION

In this paper, we proposed an effective continuous CSL recognition method, which is based on the combination of CNN and LSTM. With the powerful feature extraction of CNN, and the ability of LSTM network to learn from contextual information, this method achieved remarkable accuracy in the experiments on our self-built dataset. And the CNN-LSTM model has been proved suitable for continuous SLR without any external devices.

For the future work, we will focus on increasing recognition speed and accuracy on large datasets. And we will develop a real-time system based on this approach. We believe that the proposed method can meet the actual application needs and the real-time system will support the communication between the deaf-mute and the ordinary.

REFERENCES

[1] Tamura, S. and Kawasaki, S., “Recognition of sign language motion images,” Pattern Recognition, 21(4), 343-353 (1988).

[2] Starner, T. and Pentl, A. “Visual Recognition of American Sign Language Using Hidden Markov Models,” International Workshop on Automatic Face & Gesture Recognition, 189-194 (1995).

[3] Al-Rousan, M., Assaleh, K., and Tala’A, A., “Video-based signer-independent Arabic sign language recognition using hidden Markov models,” Applied Soft Computing, 9(3), 990-999 (2009).

[4] Chang, C.C. and Pengwu, C.M., "Gesture recognition approach for sign language using curvature scale space and hidden Markov model," 2004 IEEE International Conference on Multimedia and Expo (ICME). 2, 1187-1190(2004).

[5] Oszust, M. and Wysocki, M., [Modelling and Recognition of Signed Expressions Using Subunits Obtained by Data–Driven Approach] Springer Berlin Heidelberg, (2012).

[6] Keskin, C., Kirac, F., Kara, Y.E., and Akarun, L., "Real time hand pose estimation using depth sensors," Proceedings of the IEEE International Conference on Computer Vision, 1228-1234(2011).

[7] Mekala, P., Gao, Y., Fan, J., and Davari, A., “Real-time sign language recognition based on neural network architecture,” Proceedings of the Annual Southeastern Symposium on System Theory, 10(11), 195-199 (2011).

[8] Sawant, S.N. and Kumbhar, M.S., "Real time Sign Language Recognition using PCA," Proceedings of 2014 IEEE International Conference on Advanced Communication, Control and Computing Technologies, ICACCCT 2014, 1412-1415(2014).

[9] GAO, W., MA, J., WU, J., and WANG, C., “SIGN LANGUAGE RECOGNITION BASED ON HMM/ANN/DP,” International Journal of Pattern Recognition & Artificial Intelligence, 14(5), 587-602 (2011).

[10] Zhang, X., Chen, X., Li, Y., Lantz, V., Wang, K., and Yang, J., “A Framework for Hand Gesture Recognition Based on Accelerometer and EMG Sensors,” IEEE Transactions on Systems Man & Cybernetics Part A Systems & Humans, 41(6), 1064-1076 (2011).

[11] J. Zhang, H. Lin, and M. Zhao, "A Fast Algorithm for Hand Gesture Recognition Using Relief," 6th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2009. 1, 8-12 (2009).

[12] Graves, A., [Supervised Sequence Labelling with Recurrent Neural Networks] Springer Berlin Heidelberg, (2012).

[13] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P., “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, 86(11), 2278-2324 (1998).

[14] Krizhevsky, A., Sutskever, I., and Hinton, G.E., “ImageNet classification with deep convolutional neural networks,” Advances in Neural Information Processing Systems, 25(2), (2012).

[15] Zeiler, M.D., “ADADELTA: An Adaptive Learning Rate Method,” Computer Science, (2012).

[16] Tieleman, T. and Hinton, G. E., Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. In Coursera: Neural Networks for Machine Learning, (2012).

[17] Akyol, S. and Alvarado-Moya, P., "Finding Relevant Image Content for mobile Sign Language Recognition."IASTED International Conference Signal Processing, 48-52(2001).

[18] “OpenCV 2.3.2 documentation,” Opencv dev team, 06 April 2012, http://www.opencv.org.cn/opencvdoc/2.3.2/html/index.html(14 December 2017)

[19] Herodotou, N., Plataniotis, K.N., and Venetsanopoulos, A.N., "A color segmentation scheme for object-based video coding," 1998 IEEE Symposium on Advances in Digital Filtering and Signal Processing. Symposium Proceedings, 25-29(1998).