SIGN LANGUAGE RECOGNITION WITH LONG SHORT-TERM MEMORY

Tao Liu, Wengang Zhou, and Houqiang Li

University of Science and Technology of China
Department of Electronic Engineering and Information Science
Hefei, Anhui, P.R. China

ABSTRACT

Sign Language Recognition (SLR) aims at translating the Sign Language (SL) into speech or text, so as to facilitate the communication between hearing-impaired people and the normal people. This problem has broad social impact, however it is challenging due to the variation for different people and the complexity in sign words. Traditional methods for SLR generally use handcrafted feature and Hidden Markov Models (HMMs) modeling temporal information. But reliable handcrafted features are difficult to design and not able to adapt to the large variations of sign words. To approach this problem, considering that Long Short-Term memory (LSTM) can model the contextual information of temporal sequence well, we propose an end-to-end method for SLR based on LSTM. Our system takes the moving trajectories of 4 skeleton joints as inputs without any prior knowledge and is free of explicit feature design. To evaluate our proposed model, we built a large isolated Chinese sign language vocabulary with Kinect 2.0. Experimental results demonstrate the effectiveness of our approach compared with traditional HMM based methods.

Index Terms— Sign Language Recognition, Recurrent Neural Network, Long Short-Term Memory

1. INTRODUCTION

Sign Language (SL) is an efficient tool for hearing impaired people to communicate with each other. However, it is too difficult for normal people to understand it without special learning. Therefore it is essential to build a system to translate sign language into text or speech automatically.

Sign language is expressed by hand-shapes, trajectories of hand joints, and even facial expressions. Recently, most methods for SLR are based on hand posture recognition (HPR) and hand-joint trajectories of sign words. For HPR, Murakami *et al.* [1] focused on Japanese sign language based on recurrent neural network with data gloves equipped. Their system could recognize finger alphabet of 42 symbols. Oz *et al.* [2] designd an alphabet posture recognition system for American Sign Language (ASL) based on artificial neural network (ANN). In addition, a real time hand posture recognition system was built by J. Huang *et al.* [3, 4] based on deep neural network. Considering about hand-joint trajectory information, Lin *et al.* [5] proposed a curve matching method for sign language recognition. Selebi *et al.* [6] proposed a weighted Dynamic Time Warping method. Grobel *et al.* [7] combined both trajectory and hand shape feature, and used data gloves in their experiment. Although SLR with data gloves achieves a high-accuracy recognition, data gloves may be inconvenient in real SLR systems. Therefore, vision based method are attracting more attention[8, 9, 10, 11, 12].

The development of SLR is significantly boosted by the advanced sensor Kinect developed by Microsoft [13], which can capture both color and depth information and get joint location accurately. Some researchers focused on skeleton feature, Pu *et al.* [14] proposed a method based on trajectories and achieved an accuracy of 67.3%, while Sun *et al.* [15] proposed a discriminative exemplar coding method based on joint location. Some other researchers integrated skeleton feature and hand posture feature to realize SLR. Wang *et al.* [16] proposed a Light-HMMs method based on both HOG and skeleton feature and achieve an 84.2% performance. Sun *et al.* [17] proposed a latent support vector machine method and obtained an accuracy of 86.0% with 73 classes of ASL.

To capture the sequential dynamics of sign languages, Recurrent Neural Network (RNN) is a promising tool for this task. While RNN can well model temporal sequences, the problem of vanishing gradient and error blowing up problem [18] makes it hard to train RNN when the task contains delays of more than about 10 time steps between relevant input and target events. In [19], Donahue *et al.* showed that LSTM can provide significant improvement when ample training data was available to learn or refine the representation. The authors proposed a Long-term Recurrent Convolutional (LRCN) model that can be explored in three applications, *i.e.*, activity recognition, image description, and video description. Du et al. [20] proposed an end-to-end hierarchical RNN for skeleton based action recognition. Moreover, LSTM was also successfully used on speech recognition [21] and machine translation [22].

In this paper, we propose an end-to-end SLR system based on skeleton joint trajectories with LSTM. We only use the

skeleton joints provided by Kinect without color and depth information. The rest of this paper is organized as follows: we first describe our method in Section 2. Then we discuss the experimental results in Section 3. Finally, in Section 4, we make a conclusion and a brief discussion for future work.

2. OUR METHOD

We first review some key points about RNN and LSTM, and describe how LSTM refines the RNN and overcomes the vanishing gradient problem in Section 2.1. Then we introduce the architecture based on LSTM to solve the problem of SLR in Section 2.2.

2.1. Background of RNN and LSTM

Different from conventional ANN which directly maps the recurrent input to output, RNN also relies on previous inputs [23]. It means that RNNs remember the information of previous inputs and model the temporal sequence. Fig. 1 illustrates a RNN unit (left) and an LSTM memory block with a single cell.

A RNN with input sequence $\boldsymbol{x} = (x_0, \ldots, x_{T-1})$ of length T , H hidden states $\boldsymbol{h} = (h_0, \ldots, h_{H-1})$, and Y outputs $\boldsymbol{y} = (y_0, \ldots, y_{Y-1})$ can be defined as follows:

$$h_t = \theta_c(W_{xh}x_t + W_{hh}h_{t-1} + b_h), \quad (1)$$

$$y_t = \theta_y(W_{hy}h_t + b_y), \quad (2)$$

where W_{xh}, W_{hh}, and W_{hy} are the connection weights from the input layer x to the hidden layer h, the hidden layer h to itself and the hidden layer h to the output layer y, respectively, while θ_c and θ_y are the non-linearity activation function in the hidden layer and the output layer, respectively, and b_h and b_y are two biases.

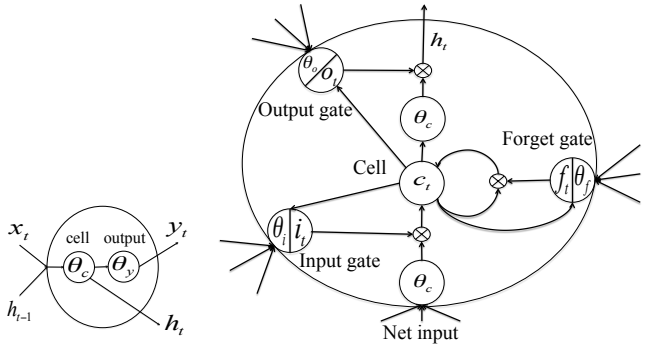

Fig. 1. A RNN cell (left) and an LSTM block with one cell (right) [23]. The LSTM block contains one self-connected memory cell and three multiplicative units: the input gate, the output gate, and the forget gate.

Unfortunately, due to the vanishing gradient and error blowing problems, it is hard to train an RNN for tasks containing long-term delay between inputs and outputs. LSTM provides a solution by replacing the RNN units with LSTM blocks. In LSTM, each block has a memory of previous network status, and flexibly update hidden states and forget previous hidden states. Each memory block contains one self-connected memory cell and three multiplicative units: the input gate i, the output gate o, and the forget gate f. Given an input, the LSTM updates as follows [23]:

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

$$f_t = \theta_f(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1}), \quad (4)$$

$$c_t = f_t c_{t-1} + i_t \theta_c(W_{xc}x_t + W_{hc}h_{t-1}), \quad (5)$$

$$o_t = \theta_o(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t), \quad (6)$$

$$h_t = o_t \theta_t(c_t), \quad (7)$$

where all the matrices W represent the connection weights between two units, all the function θ represent the non-linearity activation functions, i_t, f_t, c_t, o_t, h_t represent outputs of input gate, forget gate, cell, output gate, and block at time t, respectively.

2.2. Our LSTM for SLR

We adopt Microsoft Kinect 2.0 as the sensor of sign language and extract skeleton joint locations to represent sign words. Although Kinect can provide 25 skeleton joint locations, some of them (e.g., head, shoulder, legs) contain little useful information for SLR. In accordance with some other works such as [14], we retain 4 skeleton joints in our method: left hand, right hand, left elbow, and right elbow. Different from traditional trajectory based methods using handcrafted feature and hidden Markov models (HMMs) to model the motion information, we design a LSTM based end-to-end model for SLR.

Based on the LSTM introduced above, a variety of architectures can be devised. In the following, we describe the architecture of our LSTM designed for SLR. The framework of our model is shown in Fig 2. Our model consists of seven layers including the input layers. The first layer is the input layer which is fed with a 12-dimensional feature vetctor comprised by four 3D spatial coordinate vectors that represent the skeleton joints. The next layer is an LSTM layer with 512 dimensions. The inputs of the LSTM layer at time t are the outputs of the input layer at time t (x_t) and the outputs of the LSTM layer at time $t-1$ (h_{t-1}). After the LSTM layer, we have two fully connected layers. The first fully connected layer contains 512 neurons and the second contains 100 neurons that correspond to 100 classes. Then we set up a softmax layer followed by a pooling layer. The last layer is the output layer predicting the class of the sequence.

Training our LSTM model contains forward pass and backward pass. In the forward pass stage, the LSTM layer maps input x_t and previous timestep hidden state h_{t-1} to update hidden state h_t. Then the hidden state pass though the two fully connected layers. Finally, softmax layer takes

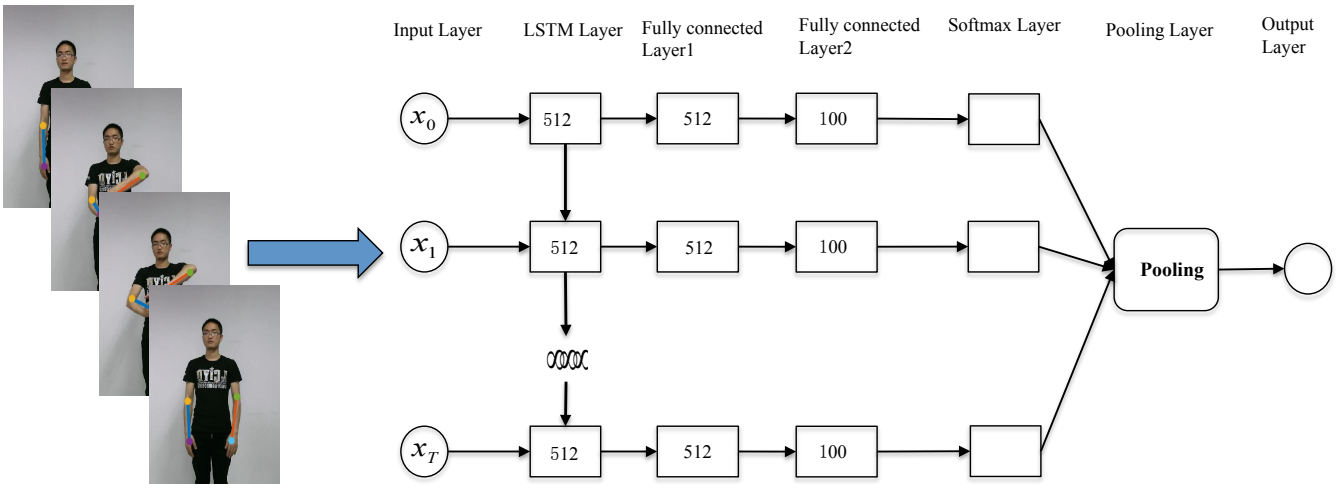

Fig. 2. Our LSTM architecture for SLR. This architecture consists of 4 skeleton joints as input, one LSTM layer, two fully connection layer, one softmax layer, one pooling layer and one output layer. Details of the architecture are described in the text.

the outputs y_t of fully connected layers as input, predict a distribution $p(y_t = s)$ at timestep t by Eq. 8, and calculate the loss by Eq. 9 [23]:

$$p(y_t = s) = \frac{\exp(y_{t,s})}{\sum_{s' \in S} \exp(y_{t,s'})}, \quad (8)$$

$$J(\boldsymbol{x}) = -\sum_{t=0}^{T-1} \ln \sum_{k=0}^{S-1} \delta(k-s) p(S_k|x_t), \quad (9)$$

where S represents category number of sign words, $\delta(\cdot)$ is Kronecker funtion, s is the groundtruth label of sign word, and $\boldsymbol{x}$ is the input sequence with a arbitrary length T . The cost function of our model is to minimize the maximum-likelihood loss function. In the backword pass, we use stochastic gradient descent to minimize the loss function and back-propagation through time algorithm to update all the weights.

In the recognition stage, our model maps each timestep of a sign word to a distribution over S classes in softmax layer without calculating the loss. Then we pool all the distribution of a sign word and predict the label of this sign word.

3. EXPERIMENTS

In this section, we first introduce our sign language data set built by Microsoft Kinect. Then we evaluate the effectiveness of our LSTM model on this dataset in comparison with other related work.

3.1. Datasets and Settings

We build two Kinect sign language datasets by ourselves and will release it to the public in the future. Dataset I contains 100 isolated Chinese sign language words that are widely used in daily life. 50 signers play each word for 5 times. So each word has 250 samples, and the dataset consists of 25,000 samples. To show the recognition performance on large vocabulary dataset, we build dataset II, which is composed of 500 sign words by 50 signers with 5 repetitions and 125,000 samples in total. We divide each dataset into 2 subsets, one for training and another for testing. In the training subsets, we randomly choose 36 signers from all the 50 signers in both datasets. The rest of the dataset constitutes the testing subsets. Details of our datasets are shown in Table 1.

Datasets		Signs	Signers	Repetitions	Samples
I	Training	100	36	5	18000
	Testing	100	14	5	7000
II	Training	500	36	5	90000
	Testing	500	14	5	35000

Table 1. The details of our datasets

The data is recorded by Kinect 2.0, and we can capture color image, depth map, and skeleton joint location in real-time. In this work, we ignore the color and depth information, and only focus on exploring the trajectory of four skeleton joints.

3.2. Impact of Parameters

We study the impact of the parameters in dataset I and obtain the optimal ones. Then we use the same parameters on dataset II. One of the most important parameters is the number of hidden units for LSTM layer. We vary this parameter from 256 to 1024. As shown in Fig. 3, we can obtain the best result if setting 512 hidden units, which is used in the following experiments.

Then we test different numbers of nodes for the first fully

connected layer. We vary this parameter from 256 to 1024. As shown in Fig. 4, the performance first grows to a peak and declines gradually, with the best result achieved when the node number is equal to 512. As for the second fully connected layer, it consists of 100 units corresponding to 100 different sign words in the experiment of dataset I and 500 units in the experiment of dataset II.

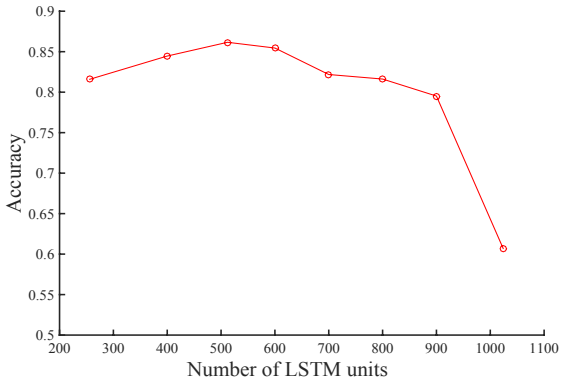

Fig. 3. Recognition accuracy with various numbers of LSTM units.

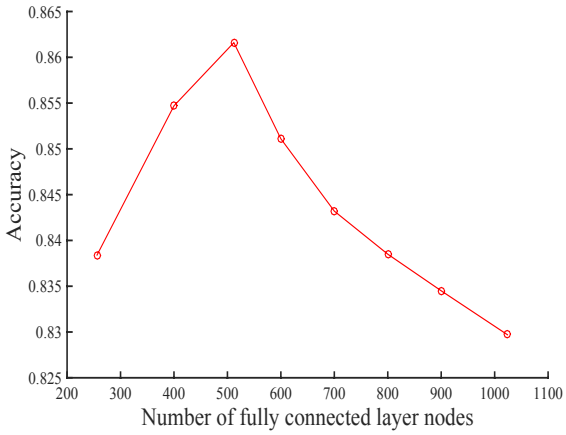

Fig. 4. Recognition accuracy with various numbers of first fully connected layer units.

3.3. Results and Analysis

After fixing the parameters for the network, we use the dataset built by ourselves to evaluate the performance of our method. We implement the method proposed in [5] which matches curve from the view of manifold (CM_VoM) and method proposed in [14] which is based on trajectory model with HMM (TM_HMM) . Besides, we include another experiment that we take normal skeleton joints as input and use normal HMM modeling the temporal information. We also test a variant of our architecture that has only one fully connected layer (LSTM_fc1). The recognition accuracy rates of the methods above and our method in Dataset I are shown in Table 2. We can see from the table that our method dramatically outperforms the compared methods, and it is noted that with two fully connected layers, the performance is better than that with only one fully connected layer. It proves that our LSTM architecture has the capability to learn contextual information of sign language and end-to-end model is more reliable. In order to show the effectiveness and stability of our method, we also conduct experiments on the larger Dataset II, and conclude the accuracies of different methods in Table 3. Our method can get a recognition rate of 63.3%, which performs better than the other methods. It shows that the end-to-end model is more reliable on large sign language datasets.

Method	Features	Accuracy
Normal HMM	Normal skeleton joints	0.332
CM_VoM [5]	MLS	0.576
TM_HMM [14]	Shape Context	0.673
LSTM_fc1	Normal skeleton joints	0.856
LSTM_fc2	Normal skeleton joints	0.862

Table 2. The accuracy of different methods on Dataset I

Method	Features	Accuracy
Normal HMM	Normal skeleton joints	0.119
CM_VoM [5]	MLS	0.514
TM_HMM [14]	Shape Context	0.502
LSTM_fc1	Normal skeleton joints	0.620
LSTM_fc2	Normal skeleton joints	0.633

Table 3. The accuracies of different methods on Dataset II

4. CONCLUSION AND FUTURE WORKS

In this paper, we propose an end-to-end model based on LSTM for Chinese sign language recognition. Our model learns temporal information of sign language by LSTM automatically and is free of designing handcrafted features. The experiments show that our model outperforms the compared methods by a large margin on large datasets. For the future work, we will combine hand-joint trajectories of sign words with hand shapes to improve the SLR performance.

5. ACKNOWLEDGEMENT

This work was supported by NSFC under contract No. 61325009, 61272316, and 61472378, Anhui Provincial Natural Science Fundation under contract No. 1508085MF109, and the Fundamental Research Funds for the Central Universities. This work is also supported by a donation of Tesla K40 GPU from NVIDIA Corporation.

6. REFERENCES

[1] K. Murakami and H. Taguchi, "Gesture recognition using recurrent neural networks," in *Proceedings of the SIGCHI Conference on Human Cactors in Computing Systems*. ACM, 1991, pp. 237–242.

[2] C. Oz and M. C. Leu, "Recognition of finger spelling of american sign language with artificial neural network using position/orientation sensors and data glove," in *Advances in Neural Networks*, pp. 157–164. Springer, 2005.

[3] J. Huang, W. Zhou, H. Li, and W. Li, "Sign language recognition using real-sense," in *IEEE China Summit and International Conference on Signal and Information Processing*, 2015, pp. 166–170.

[4] A. Tang, K. Lu, Y. Wang, J. Huang, and H. Li, "A real-time hand posture recognition system using deep neural networks," *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 2, pp. 21, 2015.

[5] Y. Lin, X. Chai, Y. Zhou, and X. Chen, "Curve matching from the view of manifold for sign language recognition," in *Computer Vision-Asian Conference on Computer Vision Workshops*. Springer, 2014, pp. 233–246.

[6] S. Celebi, A.S. Aydin, T.T. Temiz, and T. Arici, "Gesture recognition using skeleton data with weighted dynamic time warping.," in *International Conference on Computer Vision Theory and Applications*, 2013, pp. 620–625.

[7] K. Grobel and M. Assan, "Isolated sign language recognition using hidden markov models," in *IEEE International Conference on Systems, Man, and Cybernetics*, 1997, vol. 1, pp. 162–167.

[8] J. Huang, W. Zhou, H. Li, and W. Li, "Sign language recognition using 3d convolutional neural networks," in *IEEE International Conference on Multimedia and Expo*, 2015, pp. 1–6.

[9] J. Zhang, W. Zhou, and H. Li, "A new system for chinese sign language recognition," in *IEEE China Summit and International Conference on Signal and Information Processing*, 2015, pp. 534–538.

[10] J. Zhang, W. Zhou, and H. Li, "Chinese sign language recognition with adaptive hmm," *EEE International Conference on Multimedia and Expo*, 2016.

[11] J. Zhang, W. Zhou, and H. Li, "A threshold-based hmm-dtw approach for continuous sign language recognition," in *Proceedings of International Conference on Internet Multimedia Computing and Service*. ACM, 2014, p. 237.

[12] W. Zhou, M. Yang, H. Li, X. Wang, Y. Lin, and Q. Tian, "Towards codebook-free: Scalable cascaded hashing for mobile image search," *IEEE Transactions on Multimedia*, vol. 16, no. 3, pp. 601–611, 2014.

[13] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore, "Real-time human pose recognition in parts from single depth images," *Communications of the ACM*, vol. 56, no. 1, pp. 116–124, 2013.

[14] J. Pu, W. Zhou, J. Zhang, and H. Li, "Sign language recognition based on trajectory modeling with hmms," in *MultiMedia Modeling*. Springer, 2016, pp. 686–697.

[15] C. Sun, T. Zhang, B. Bao, C. Xu, and T. Mei, "Discriminative exemplar coding for sign language recognition with kinect," *IEEE Transactions on Cybernetics*, vol. 43, no. 5, pp. 1418–1428, 2013.

[16] H. Wang, X. Chai, Y. Zhou, and X. Chen, "Fast sign language recognition benefited from low rank approximation," in *IEEE International Conference and Workshops on Automatic Face and Gesture Recognition*, 2015, vol. 1, pp. 1–6.

[17] Chao Sun, Tianzhu Zhang, Bing-Kun Bao, and Changsheng Xu, "Latent support vector machine for sign language recognition with kinect," in *IEEE International Conference on Image Processing*, 2013, pp. 4190–4194.

[18] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[19] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," *arXiv preprint arXiv:1411.4389*, 2014.

[20] Y. Du, W. Wang, and L. Wang, "Hierarchical recurrent neural network for skeleton based action recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1110–1118.

[21] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *Proceedings of International Conference on Machine Learning*, 2014, pp. 1764–1772.

[22] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, 2014, pp. 3104–3112.

[23] A. Graves, *Supervised sequence labelling with recurrent neural networks*, vol. 385, Springer, 2012.