CapsNet-Based Chinese Fingerspelling Recognition Postprint

Authors: Hao Ziyu, Alip Kurban, Li Xiaohong, Isa Uatibek

Date: 2018-08-13T00:00:00+00:00

Abstract

Recognition of Chinese finger spelling constitutes an important component of Chinese Sign Language recognition, facilitating more convenient communication for hearing-impaired individuals and human-computer interaction. Traditional finger spelling recognition employs Convolutional Neural Network methods with simplistic model structures, where pooling layers discard substantial information. Capsule, a sub-network constructed and abstracted within neural networks, enables each capsule to focus on individual tasks while preserving spatial features of images. This study analyzes the characteristics of finger spelling in Chinese Sign Language, constructs and expands a finger spelling image training dataset, attempts to address finger spelling recognition tasks using the CapsNet (Capsule Network) model, compares the recognition accuracy of CapsNet under different parameters, and contrasts it with the classic GoogLeNet convolutional network. Experimental results demonstrate that CapsNet can achieve favorable recognition performance on sign language recognition tasks.

Full Text

Preamble

Chinese Finger Language Recognition Using CapsNet

Hao Ziyu, Alifu · Kuerban†, Li Xiaohong, Yisa · Wuhatibieke

(School of Software, Xinjiang University, Urumqi 830046, China)

Abstract: As a crucial component of Chinese sign language recognition, Chinese finger language recognition facilitates communication for the hearing-impaired and enhances human-computer interaction. Traditional finger language recognition employs convolutional neural networks (CNNs) with simplistic architectures that discard substantial information during pooling.

Capsules are constructed subnetworks within neural networks where each capsule focuses on specific tasks while preserving spatial features of images. This paper analyzes the characteristics of finger language in Chinese sign language, constructs and expands a finger language image training set, and attempts to solve finger language recognition tasks using the CapsNet model. We compare CapsNet recognition rates under different parameters and against the classic GoogLeNet convolutional network. Experimental results demonstrate that CapsNet achieves superior recognition performance in sign language recognition tasks.

Keywords: sign language; finger language recognition; neural network; CapsNet

0 Introduction

Sign language serves as the primary tool for deaf and hard-of-hearing individuals to exchange ideas and interact socially, playing a vital role in knowledge acquisition, personal development, and social cognition within this community [1]. Sign language recognition employs pattern recognition techniques to analyze motion and posture features of hands and arms, using sequential features as classifier inputs for classification tasks that ultimately translate sign language into text or speech output, thereby facilitating daily communication for the hearing-impaired [2]. Additionally, sign language recognition provides convenient conditions for hearing individuals to learn and comprehend sign language.

Finger spelling constitutes an essential component of Chinese sign language, enhancing its completeness. By integrating finger spelling, the expressive methods of gesture language are improved, making gestures more precise and enriched. The *Chinese Finger Spelling Alphabet Scheme* specifies 30 letter gestures, as shown in Figure 1 [Figure 1: see original paper]. Gu Dingqian et al. [3] argue that using finger spelling to express specific sign language meanings is a common phenomenon in Chinese sign language. Finger spelling functions not only as a morpheme but also frequently as a basic word, manifesting in three primary forms:

a) **Single-letter gestures:** These use a single finger spelling gesture representing an initial consonant to denote an entire word. Except for the letter "v," all 29 other finger spelling gestures have appeared independently as basic words, such as the signs for "bìlǜ" (jade green) and "bái" (white), illustrated in Figure 2 [Figure 2: see original paper].

b) **Modified-letter gestures:** These basic words still employ finger spelling but with variations in usage, hence termed "modified forms." For example, one or two finger spelling gestures with changed orientation, additional movements, or placed at specific body locations represent a word, such as the sign for "Féng."

c) **Syllabic gestures:** These use complete finger spelling gestures represent-

ing both initial and final syllables to denote a word, such as the sign for "Wú." The gestures for "Féng" and "Wú" are shown in Figure 3 [Figure 3: see original paper].

Evidently, finger spelling is indispensable in Chinese sign language, capable of expressing sign language information either independently or accompanied by one or two gesture movements. Finger spelling makes sign language expression more accurate and enriched, particularly beneficial for expressing uncommon words and phrases.

On one hand, sign language recognition can serve as a translation tool between hearing individuals and deaf individuals, providing better services for the deaf community; on the other hand, as part of human language understanding, sign language recognition can function as a means of human-computer interaction. In summary, finger language recognition constitutes an important foundation and component of sign language recognition, holding significant importance for Chinese sign language recognition [4].

1 CapsNet Model

Current computer vision-based sign language recognition methods primarily employ convolutional neural networks [5]. CNNs demonstrate outstanding performance in image classification tasks. A complete convolutional network typically includes input layers, convolutional layers, pooling layers, fully connected layers, and output layers. Traditional CNNs build upon the LeNet5 model, improving performance by increasing network depth, adding network width, or changing activation functions [6]. LeNet5 was the first successful convolutional neural network applied to digit recognition, with its structure shown in Figure 4 [Figure 4: see original paper].

CNNs leverage three key characteristics—local receptive fields, shared weights, and spatial/temporal pooling for downsampling—to fully utilize locality features inherent in data while ensuring translation invariance to some degree [7]. The weight-sharing structure of convolutional models resembles biological neural networks, reducing network complexity and decreasing the number of weights. This structural characteristic makes them particularly suitable for machine learning with large image data, enabling continuous dimensionality reduction for massive image recognition problems, where pooling structures significantly improve computational efficiency. However, precisely because of such pooling structures, effective data information is lost during feature map sampling. Deep learning pioneer Hinton argued that pooling solves the wrong problem—we should organize information rather than discard it—and proposed Capsule theory [8].

CapsNet comprises n subnetworks (Capsules), where each capsule focuses on specific tasks while preserving spatial features of images. CapsNet uses activation vectors to represent both the presence of an entity and its attributes. Different values across vector dimensions represent different attributes, while the vector' s magnitude represents the probability of entity occurrence. To en-

sure the vector length (i.e., the probability of entity occurrence) falls between 0 and 1, the vector undergoes nonlinear compression and normalization, with its direction in high-dimensional space reflecting different attributes of the entity. The squashing nonlinear function guarantees the output vector length remains between 0 and 1. The squashing function is expressed as:

$$v_j = \frac{\|s_j\|^2}{1 + \|s_j\|^2} \cdot \frac{s_j}{\|s_j\|}$$

where v_j is the output vector of a Capsule and s_j is the weighted sum of vectors output from the previous layer of Capsules. This nonlinear function preserves the direction of input vectors while compressing their length within the interval [0,1). The input to a vector occurs in two stages, as follows:

$$s_j = \sum_i c_{ij} \cdot \hat{u}_{j|i}$$

where $\hat{u}_{j|i}$ is derived from the product of lower-level Capsule outputs u_i and weight matrices; c_{ij} are coupling coefficients from the dynamic routing process, calculated as:

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_k \exp(b_{ik})}$$

The consistency between vectors is measured using the inner product between prediction vectors $\hat{u}_{j|i}$ and output vectors v_j to update b_{ij}. Softmax is used to update coupling coefficients, further refining the input s_j to the next layer of Capsules, ultimately producing new v_j. This process iteratively updates consistency parameters. The hierarchical structure between Capsules is shown in Figure 6 [Figure 6: see original paper].

This method continuously updates b_{ij} without requiring backpropagation algorithms. Moreover, the Routing algorithm converges easily, achieving satisfactory results after just three iterations. However, other convolutional parameters and weight matrices W_{ij} within Capsules still require updating based on the loss function, typically using standard backpropagation.

As shown in Figure 5 [Figure 5: see original paper], the CapsNet model also employs convolutional structures to extract features, but Primary Caps (the Capsule preparation layer) can divide data information into multiple units across multiple channels, generating spatial-information-preserving vectors for each unit that are then input into the next layer of Capsule neurons. This structure replaces traditional pooling layers in convolutional networks, effectively reducing information loss. The final layer resembles a fully connected layer, but each neuron is transformed into a Capsule structure for classification output, called the DigitCaps layer.

2.1 Finger Language Acquisition and Preprocessing

Kinect is a depth camera released by Microsoft [9], whose depth data and human skeletal point data have opened broader avenues for gesture recognition research. Hand images can be extracted using Kinect's BodyIndex method to obtain depth images [10].

Since Kinect depth values deviate with increasing distance, the hand being captured was positioned 1.2–1.5 meters directly in front of the Kinect for optimal experimental results. Data was collected from three angles: directly in front, left side, and right side of the gesture. The resulting binary finger language images place the hand at the image center and resize them to 44$×$44 binary images to reduce computational load.

2.2 Image Filtering

Traditional filtering algorithms include mean filtering, median filtering, and Gaussian filtering. Mean filtering is relatively simple and easy to implement but blurs object edges and is sensitive to zero-value noise, affecting subsequent processing. Gaussian filtering's smoothing effect depends on standard deviation, with higher weights for pixels closer to the center, yielding better smoothing results. Since noise in depth images captured by Kinect mostly consists of zero-value points (where the camera cannot obtain depth values), median filtering can effectively remove noise points while preserving hand edge information [11]. Therefore, this paper adopts median filtering for denoising.

Median filtering is a nonlinear smoothing technique whose basic principle involves using a template operator to sort all pixel values within a covered region, updating the current pixel value with the median of these pixel points. The median filtering template size used in this paper is 3×3.

2.3 Dataset Expansion

When training deep neural networks for image recognition, using large amounts of training data improves network performance, such as increasing classification accuracy and preventing overfitting. Acquiring more training samples is costly and often difficult to achieve in practice, but artificially expanding training data [12] can achieve similar effects.

This paper expands the finger language image dataset using horizontal flipping, rotation, and adding random salt-and-pepper noise on three directional finger language images. These methods simulate real-world variations and enhance model accuracy and generalization capability [13].

We collected finger language for the first 10 letters (a–j) of the Chinese finger alphabet. After processing all data, we generated 6,500 finger language images, randomly selecting 500 as the test set. The data processing workflow is shown in Figure 7 [Figure 7: see original paper].

3 Experiments

The experimental hardware environment consisted of an Intel Core i5-3230 CPU @ 2.60 GHz quad-core processor, 8 GB RAM, and an NVIDIA GeForce GT 645M 2 GB graphics card. The input was 44$×$44 image data. Due to the numerous finger language types and complex hand shape contours, larger convolutional kernels could extract more hand shape edge features. In this experiment, the first two convolutional kernel sizes were set to 8×8, 10×10, and 13×13 for comparative experiments under different configurations [14]. The first convolutional layer had a stride of 1×1, the second convolutional layer had a stride of 2×2, the backpropagation algorithm iterated 20 epochs, and the dynamic routing algorithm iterated 3 times.

Initially, the convolutional kernel size was set to 10×10, all images underwent median filtering, and experiments were conducted without added noise, achieving high recognition rates. The loss function variation is shown in Figure 8 [Figure 8: see original paper]. The figure demonstrates that the CapsNet model can quickly fit the data, stabilizing after 4,500 training steps. The training effectiveness is shown in Figure 9 [Figure 9: see original paper]. Under these experimental conditions, the CapsNet model achieved a 95.8% recognition rate on the test set.

After adding random salt-and-pepper noise to the above experiments, the recognition rate decreased insignificantly, proving the model' s robustness. We repeated experiments using 8×8 and 13×13 convolutional kernels, with comparative results shown in Table 1 .

Table 1. Comparison of average recognition rates under different parameters

Kernel Size	Average Recognition Rate
8×8	92.6%
10×10	95.8%
13×13	95.8%

On this basis, we verified the accuracy of each finger language letter on the test set, achieving expected results. Since finger language data was collected from three angles (front, left side, right side), some gestures in letters "g" and "i" appeared similar, resulting in less ideal recognition rates of 84% and 82% respectively on the test set, while letter "j"achieved 100% recognition. Individual finger language recognition rates are shown in Figure 10 [Figure 10: see original paper].

Compared with AlexNet and VGG, GoogLeNet' s Inception structure increases network width, has fewer parameters, maintains network sparsity, and utilizes dense matrices to significantly improve computational performance [15].

Inception-v4 [16] not only incorporates characteristics from the first four Inception versions but also combines ResNet to further reduce error rates. Consequently, we conducted comparative experiments using Inception-v4 under noisy conditions. The comparison between CapsNet and Inception-v4 is shown in Table 2 .

Table 2. Experimental comparison between CapsNet and Inception-v4

Model	Average Accuracy	Single Highest Accuracy
CapsNet	95.4%	100%
Inception-v4	94.4%	100%

Table 2 shows that Inception-v4 achieved an average accuracy of 94.4%, slightly lower than CapsNet. For individual finger language letter recognition, both models reached a maximum of 100%.

4 Conclusion

This paper employs the CapsNet model using convolutional structures and dynamic routing parameter update algorithms for static finger language recognition. Experiments demonstrate that the CapsNet model achieves excellent results in finger language recognition tasks, with a highest average recognition rate of 95.4% under noisy conditions. Particularly noteworthy is the high accuracy achieved despite partial finger language features being less distinct due to three-angle data collection, indicating that this dynamic routing parameter update algorithm and vector-based property prediction approach exhibit favorable performance.

Sign language comprises four features: hand shape, orientation, location, and movement trajectory. Finger language represents only part of hand shape. Future research will focus on complete sign language recognition by combining multiple sign language features in dynamic sign language using spatiotemporal algorithms integrated with the advantageous characteristics of Capsules.

References

[1] Yu Xiaoting, He Huizhong. A review on domestic sign language study [J]. Chinese Journal of Special Education, 2009 (4): 36-41.

[2] Yao Dengfeng, Jiang Minghu, Abudoukelimu Abulizi, et al. A survey of Chinese sign language processing [J]. Journal of Chinese Information Processing, 2015, 29 (5): 216-228.

[3] Gu Dingqian, Song Xiaohua, Yu Yuanyuan. The analysis of Chinese sign language' s basic words (basic movements) [J]. Chinese Journal of Special Education, 2005 (2): 65-72.

[4] Li Yong, Gao Wen, Yao Hongxun. Chinese sign language finger alphabet recognition based on color gloves [J]. Computer Engineering & Applications, 2002, 38 (17): 55-58.

[5] Yi Jingguo, Cheng Jianghua, Ku Xishu. Review of gestures recognition based on vision [J]. Computer Science, 2016, 43 (s1): 103-108.

[6] Zhou Feiyan, Jin Linpeng, Dong Jun. Review of convolutional neural network [J]. Chinese Journal of Computers, 2017, 40 (6): 1229-1251.

[7] Li Xudong, Ye Mao, Li Tao. Review of object detection based on convolutional neural networks [J]. Application Research of Computers, 2017, 34 (10): 2881-2886, 2891.

[8] Sabour S, Frosst N, Hinton G E, et al. Dynamic routing between capsules [C]// Advances in Neural Information Processing Systems. 2017.

[9] Liu Jia, Zheng Yong, Zhang Xiaorui, et al. Overview of hand gesture tracking based on Kinect [J]. Application Research of Computers, 2015, 32 (7): 1921-1925.

[10] Deng Rui, Zhou Lingling, Ying Rendong. Gesture extraction and recognition research based on Kinect depth data [J]. Application Research of Computers, 2013, 30 (4): 1263-1265.

[11] Ruan Qiuqi. Digital image processing [M]. Beijing: Electronic Industry Press, 2007.

[12] Krizhevsky A, Sutskever I, Hinton G E, et al. Imagenet classification with deep convolutional neural networks [C]// Advances in Neural Information Processing Systems. 2012: 1097-1105.

[13] Simard P Y, Steinkraus D W, Platt J, et al. Best practices for convolutional neural networks applied to visual document analysis [C]// Proc of International Conference on Document Analysis and Recognition. 2003: 958-962.

[14] Szegedy C, Liu Wei, Jia Yangqing, et al. Going deeper with convolutions [C]// Proc of IEEE Conference on Computer Vision and Pattern Recognition. [S. I.]: IEEE Press, 2015: 1-9.

[15] Canziani A, Paszke A, Culurciello E. An analysis of deep neural network models for practical applications [J/OL]. [2018-03-07]. https://arxiv.org/abs/1605.07678.

[16] Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, Inception-ResNet and the impact of residual connections on learning [C]// Proc of the 31st AAAI Conference on Artificial Intelligence. Palo Alto, California: AAAI Press, 2017: 4278-4284.

Note: Figure translations are in progress. See original paper for figures.