SIGN LANGUAGE RECOGNITION USING 3D CONVOLUTIONAL NEURAL NETWORKS

Jie Huang, Wengang Zhou, Houqiang Li, and Weiping Li

University of Science and Technology of China, Hefei, China
hagjie@mail.ustc.edu.cn, {zhwg, lihq, wpli}@ustc.edu.cn

ABSTRACT

Sign Language Recognition (SLR) targets on interpreting the sign language into text or speech, so as to facilitate the communication between deaf-mute people and ordinary people. This task has broad social impact, but is still very challenging due to the complexity and large variations in hand actions. Existing methods for SLR use hand-crafted features to describe sign language motion and build classification models based on those features. However, it is difficult to design reliable features to adapt to the large variations of hand gestures. To approach this problem, we propose a novel 3D convolutional neural network (CNN) which extracts discriminative spatial-temporal features from raw video stream automatically without any prior knowledge, avoiding designing features. To boost the performance, multi-channels of video streams, including color information, depth clue, and body joint positions, are used as input to the 3D CNN in order to integrate color, depth and trajectory information. We validate the proposed model on a real dataset collected with Microsoft Kinect and demonstrate its effectiveness over the traditional approaches based on hand-crafted features.

Index Terms— Sign Language Recognition, 3D Convolutional Neural Networks, Deep Learning

1. INTRODUCTION

Sign language, as one of the most widely used communication means for hearing-impaired people, is expressed by variations of hand-shapes, body movement, and even facial expression. Since it is difficult to collaboratively exploit the information from hand-shapes and body movement trajectory, sign language recognition is still a very challenging task. This paper proposes an effective recognition model to translate sign language into text or speech in order to help the hearing impaired communicate with normal people through sign language.

Technically speaking, the main challenge of sign language recognition lies in developing descriptors to express hand-shapes and motion trajectory. In particular, hand-shape description involves tracking hand regions in video stream, segmenting hand-shape images from complex background in each frame and gestures recognition problems. Motion trajectory is also related to tracking of the key points and curve matching. Although lots of research works have been conducted on these two issues for now, it is still hard to obtain satisfying result for SLR due to the variation and occlusion of hands and body joints. Besides, it is a nontrivial issue to integrate the hand-shape features and trajectory features together.

To address these difficulties, we develop a 3D CNNs to naturally integrate hand-shapes, trajectory of action and facial expression. Instead of using commonly used color images as input to networks like [1, 2], we take color images, depth images and body skeleton images simultaneously as input which are all provided by Microsoft Kinect. Kinect is a motion sensor which can provide color stream and depth stream. With the public Windows SDK, the body joint locations can be obtained in real-time as shown in Fig.1. Therefore, we choose Kinect as capture device to record sign words dataset. The change of color and depth in pixel level are useful information to discriminate different sign actions. And the variation of body joints in time dimension can depict the trajectory of sign actions. Using multiple types of visual sources as input leads CNNs paying attention to the change not only in color, but also in depth and trajectory. It is worth mentioning that we can avoid the difficulty of tracking hands, segmenting hands from background and designing descriptors for hands because CNNs have the capability to learn features automatically from raw data without any prior knowledge [3]. 3D CNNs have been applied in video stream classification recently years [2, 4, 5]. A potential concern of CNNs is time consuming. It costs several weeks or months to train a CNNs with million-scale in million videos. Fortunately, it is still possible to achieve real-time efficiency, with the help of CUDA for parallel processing.

We propose to apply 3D CNNs to extract spatial and temporal features from video stream for Sign Language Recognition (SLR). Existing methods for SLR use hand-crafted features to describe sign language motion and build classification model based on these features. In contrast, 3D CNNs can capture motion information from raw video data automatically, avoiding designing features. We develop a 3D CNNs taking multiple types of data as input. This architecture integrates color, depth and trajectory information by performing convolution and subsampling on adjacent video frames. Experimental results demonstrate that 3D CNNs can

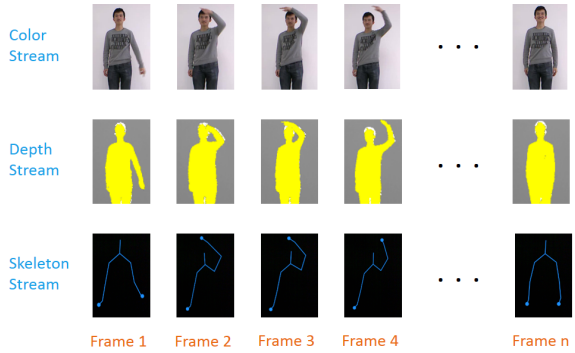

Fig. 1. Video of sign word recorded by Kinect. Video stream contains color images, depth images and images of body joints

significantly outperform Gaussian mixture model with Hidden Markov model (GMM-HMM) baselines on 25 sign words recorded by ourselves.

The rest of this paper is organized as follows: we first discuss some related works in Section 2. Then the framework and details of 3D CNNs are described in Section 3. Experimental results are discussed in Section 4. Finally, we conclude the paper in Section 5.

2. RELATED WORK

In sign language recognition, Hidden Markov Models (HMMs) have been popularly exploited. In [6], each sign is modeled with one HMM. For both training and recognition, feature vectors must be extracted from each video frame and then inputted to the HMM. Cotton gloves are used with several color-markings in signers, palm and back of the hand to get both trajectory and hand shape features from video. [7] described a HMM based system which used one color camera to track unadorned hands in real time and interpret American sign language using HMMs with a lexicon of 40 words. As an extension to HMM, Parallel Hidden Markov Mode (PaHMMs) [8] separated a sign process as several partially independent processes which can be calculated in parallel in order to address the scalability issue. They run experiments with a 22-sign vocabulary and demonstrate that PaHMMs can improve the robustness. Artificial neural networks is another widely used method for SLR. [9] used neural networks to recognize a finger alphabet of 42 symbols and then developed a recurrent neural network for gesture recognition. Huang et al. [10] presented a sign language recognition which used a 3D Hopfield neural network to recognize gesture. It achieved 91% recognition rate on 15 different hand gestures. [11] proposed a fuzzy min-max neural network taking x, y, z coordinates and angles of hands as input. This model recognized 25 isolated gestures with a recognition rate of 85%.

The standard approach to SLR involves three stages: locating regions of interest, extracting features to describe these regions, and training a classifier based on features. As a biologically inspired deep learning model, CNNs accomplish all three stages with a single framework that is trained from raw pixel values to classifier outputs. CNNs have shown significant improvement in visual object recognition [3], object detection [12], natural language processing [13], scene labeling [14] and segmentation tasks [15]. Despite of these achievements, there is little work on applying CNNs to video classification. This is partially due to the difficult in modifying the CNNs to incorporate both spatial and temporal data. [4] proposed a 3D CNNs where the third dimension corresponds to frame stamp. It took several adjacent frames as input and performs 3D convolution in the convolutional layers. [5] followed the same idea and proposed to regularize the outputs with high-level features and combines the predictions of a variety of different models. They applied the developed models to recognize human actions and achieved superior performance in comparison to baseline methods. [2] applied 3D CNNs to large scale video classification. Their spatio-temporal networks demonstrate significant performance improvements compared to strong feature-based baselines in a dataset of 1 million YouTube videos from 487 categories.

In this paper, we develop a 3D CNN to integrate multi-sources of visual data. Each type of data stream provides several adjacent frames as input. In the experiments, we show that our model outperforms the baseline method based on hand-crafted features.

3. 3D CONVOLUTIONAL NEURAL NETWORKS

First, we review some key points about CNNs which is introduced by LeCun [3]. It is known as 2D CNNs and applied widely to image domain and speech domain. Then we extend 2D convolution to 3D with time dimension and using 3D convolution to construct the architecture of our 3D CNN.

3.1. Background of 2D CNNs

In 2D CNNs, 2D convolution is performed at feature maps to extract features from the previous layer. The idea is that a kernel window called local receptive field moves over each unit from prior layer. Each unit in the convolutional layer receives inputs from a set of units located in the kernel window and calculated by equation

$$f_{xy} = tanh(\sum_{i,j} w_{ij} v_{(x+i)(y+j)} + b), \quad (1)$$

where f_{xy} is a unit in feature map at position (x, y), $v_{(x+i)(y+j)}$ is input unit at position $(x + i, y + j)$, $\tanh(\cdot)$ is an activation function, b is the bias of the feature map and w_{ij} is the weight of kernel, as shown in Fig.2. Parameters of kernel windows are forced to be identical for all its possible locations of previous layer, which is called weight shar-

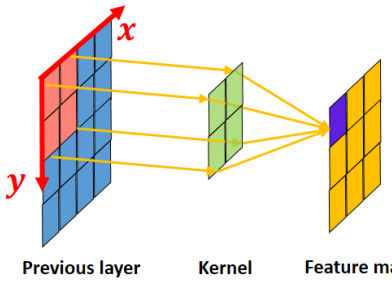

Fig. 2. 2D convolution. ($z_{00} = v_{00}w_{00} + v_{01}w_{01} + v_{10}w_{10} + v_{11}w_{11} + b$)

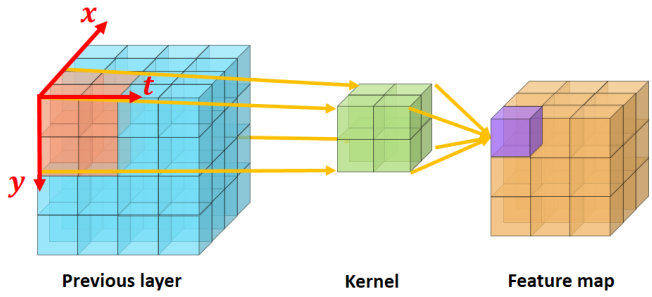

Fig. 3. 3D convolution. ($z_{000} = v_{000}w_{000} + v_{001}w_{001} + v_{010}w_{010} + v_{011}w_{011} + v_{100}w_{100} + v_{101}w_{101} + v_{110}w_{110} + v_{111}w_{111} + b$)

ing. The sharing of weights reduces the number of free variables and increases the generalization capability of the network. Weights are replicated over the input image, leading to intrinsic insensitivity to translation in the input. A convolutional layer usually contains multiple feature maps so that multiple features can be detected. In the subsampling layers, the resolution of the feature maps is reduced by pooling over local neighborhood on the feature maps in the previous layer, thereby enhancing the invariance to distortions on the inputs. A CNN architecture can be constructed by stacking multiple layers of convolution and subsampling in an alternating fashion. The network is trained with the usual back propagation gradient-descent procedure. 2D CNNs are applied on image dataset to classify them and extract spatial features.

3.2. 3D convolution

In 2D CNNs, convolutions are applied on the 2D feature maps to compute features from the spatial dimensions only. But for sign language recognition in videos, along with spatial features, it is also desirable to capture the motion information encoded in multiple contiguous frames. To effectively incorporate the motion information in video analysis, [5] proposed to perform 3D convolution in the convolutional layers of CNNs so that discriminative features along both spatial and temporal dimensions are captured. 3D convolution is achieved by convolving a 3D kernel with the cube formed by stacking multiple contiguous frames together as shown in Fig. 3. Similar to equation 1, 3D convolution is calculated by:

$$f_{xyt} = tanh(\sum_{i,j,k} w_{ijk} v_{(x+i)(y+j)(t+k)} + b), \qquad (2)$$

By this setting, the feature maps in the convolution layer are connected to multiple contiguous frames in the previous layer, thereby capturing motion information.

A 3D convolutional kernel can only extract one type of features from the frame cube, since the kernel weights are replicated across the entire cube. A general design principle of CNNs is that the number of feature maps should be increased in late layers by generating multiple types of features from the same set of lower-level feature maps. Similar to the case of 2D convolution, this can be achieved by applying multiple 3D convolutions with distinct kernels to the same location in the previous layer.

3.3. Our 3D CNN architecture

Based on the 3D convolution described above, a variety of CNN architectures can be devised. In the following, we describe a 3D CNN architecture that we have developed for sign language recognition. We use Microsoft Kinect as input device which provides color video stream, depth video stream and is capable of tracking users body movement simultaneously. Color information contains RGB three channels. In all, we obtain five types of input data in all, adding depth and body skeleton. In our CNN architecture, as shown in Fig.4, for each type of visual source we consider nine frames of size 64×48 centered on the current frame as input to the 3D CNN. That results in five feature maps denoted by color-R, color-G, color-B, depth, body skeleton. Each feature map contains nine stacked frames from the corresponding channel as a cube. Multiple feature maps as input usually lead to better performance as compared to only gray-scale intensity input.

Our architecture consists of eight layers including the input layer. After the input layer, the next four layers are convolution layers (C1) followed by sub-sampling (S1) and convolution (C2) which is further followed by sub-sampling (S2). This is followed by a 3rd convolution layer (C3) with no subsampling following. This is further followed by two fully-connected layers containing the output layer. In the architecture, it is important to design the size of kernels in different layers. First we apply 50 different 3D kernels size of 7×7×5 (7×7 in spatial dimension and 5 in the temporal dimension) on all five channels and obtain the first convolutional layer C1 which consists of 50 features map of size 58×42×5 (58×42 in spatial dimensions and 5 in temporal dimension). In the subsequent subsampling layer S1, we apply 2×2 subsampling

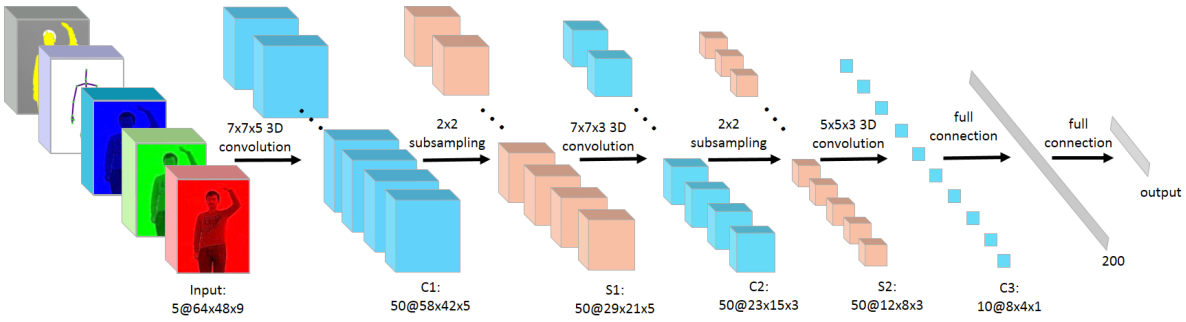

Fig. 4. Our 3D CNN architecture for sign language recognition. This architecture consists of five types of data as input, three convolution layers, two subsampling layers, and two full connection layer. Descriptions in detail are given in the text.

on each of the feature maps in the C1 layer, which results in the same number of feature maps with a reduced spatial resolution. Sub-sampling makes our model resistant to small spatial distortions. As a result of this, the subsequent layers perform pattern recognition at progressively larger spatial scales, with lower resolution. Thus a CNN with several subsampling layers enables processing of large inputs, with relatively few free weights. The second convolution layer C2 also consists of 50 feature maps of size $23\times15\times3$. This layer is obtained by applying 3D kernels of size $7\times7\times3$ followed by sub-sampling of factor 2. The third convolution layer C3 consists of 10 feature maps of size $8\times4\times1$ obtained by applying 3D kernels of size $5\times5\times3$ on layer S2.

After multiple stages of convolution and subsampling, we are able to extract spatial-temporal features from the input. 45 consecutive input frames from 5 channels have been converted into a 320D ($(8\times4\times1)\times10$) feature vector capturing the motion information in the input frames after all the convolutional layers. Now the last two fully-connected layers act as a classical multilayer perceptron classifier on the 320D input. Layer F1 consists of 200 nodes of size 1×1 and the final layer F2 (output layer) consists of 25 units corresponding to different sign words. Finally, we use back-propagation algorithm for training the model.

4. EXPERIMENTAL RESULTS

In this section, we first introduce our self-built sign language dataset collected by Kinect sensor. Then we evaluate the effectiveness of our developed 3D CNNs on this dataset in comparison with baseline method.

4.1. Dataset and settings

Currently, these is no public Kinect sign language dataset. In this situation, we built the Kinect sign language dataset by ourselves. Our dataset has 25 vocabularies that are widely used in daily life. Each word is played by 9 signers, every signer play 3 times for each word. So each word has

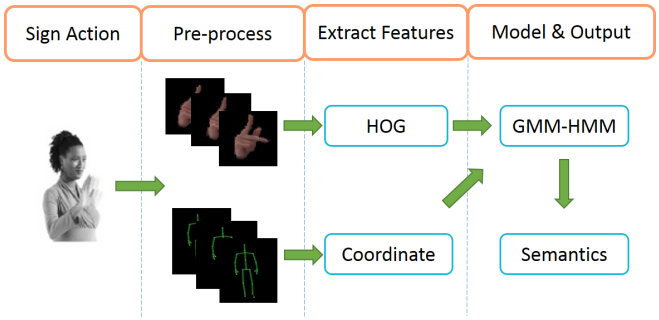

Fig. 5. GMM-HMM for sign language recognition. We use Kincet to capture sign action, then make use of color and depth information to crop hand-shape from background and depict body skeleton use joints locations. We combine two kinds of features: HOG (from hand-shape image) and coordinate locations of joints to train GMM-HMM. After training, GMM-HMM can used to output semantics of sign actions

27 samples, and we have 25×27 samples totally. In training stage, we choose 18 samples from each word randomly to train model, resulting in 25×18=450 samples as training set. The rest constitutes the testing set. The data is recorded by Kinect, capturing color image, depth map and body joints locations simultaneously.

4.2. Baseline method

Gaussian Mixture Model-Hidden Markov Model (GMM-HMM) is a conventional method in temporal pattern recognition such as, speech recognition domain and sign language recognition. Therefore GMM-HMM is regarded as a baseline method in our experiment.

As we mentioned before, CNNs are capable of learning features from raw image automatically, however GMM-HMM is not. That means we need to extract hand-crafted features from sign language video and then use those features to train GMM-HMM. On observation of sign motion, we find that both the change of hand-shape and the trajectory of body

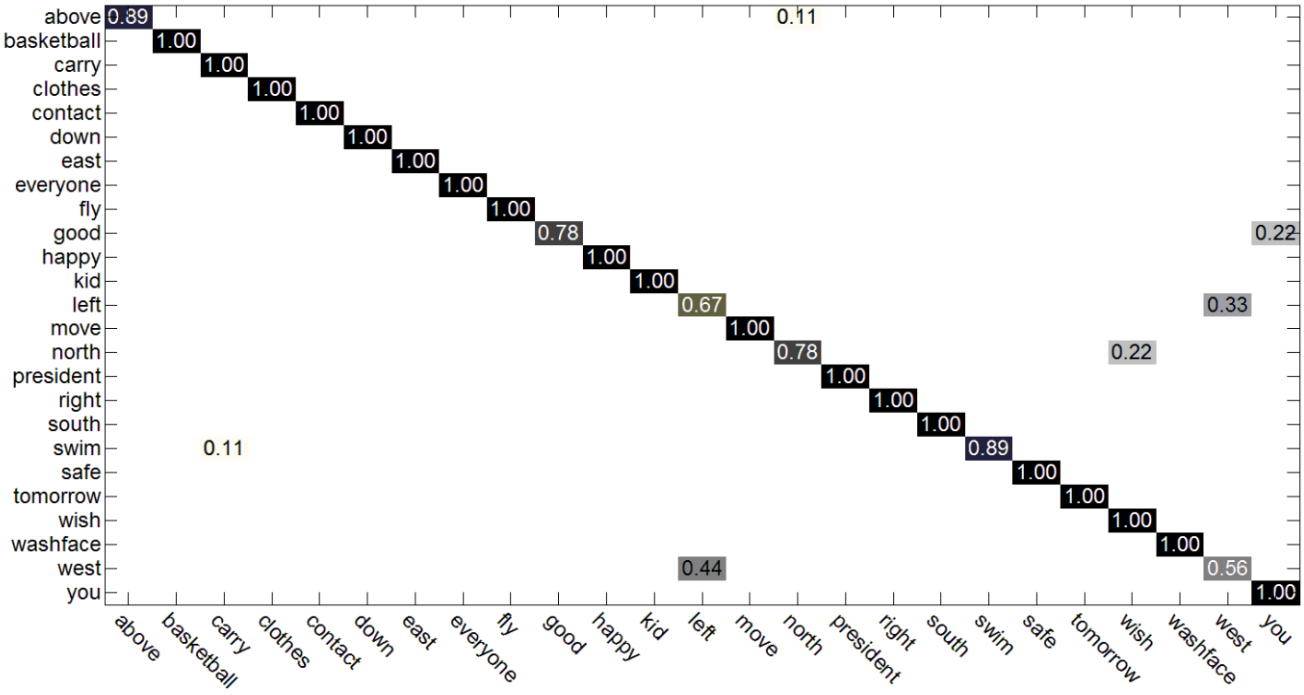

Fig. 6. Confusion matrix of our proposed method, rows correspond to actual sign action(label) and columns correspond to the recognized action

movement are two of the most important features to describe a sign motion. So we extract trajectory and hand-shape features to train GMM-HMM for recognition. The framework is shown in Fig.5. We follow the idea proposed by [16] to crop hand-shape from background. [16] also used Kinect as input device and proposed an effective algorithm to implement hand segmentation and tracking. The algorithm incorporates both color and depth information, without specific requirements on uniform-colored or stable background. We calculate 36-D HOG on 32×32 hand-shape image. The 32×32 image is regarded as one cells. Each cell has 16×16 pixels and the gradient of a cell is divided into 9 orientation bins. Every four adjacent cells constitute a block. Each image contains 1 blocks. At the same time, we use 3D coordinate position of key skeleton joints as trajectory features directly for simplicity. Skeleton joints include right hand, left hand, right wrist, left wrist, right elbow, left elbow, right shoulder, left shoulder, shoulder center and head. We obtain a 3×10=30 dimension vector as trajectory features. After combining these two kinds of features, we obtain 66-D vector as final features and used for training GMM-HMM. For recognition stage, we also extract 66-D vector from each frame of video and use the trained GMM-HMM to output semantics.

One the other hand, we also compare to the sequential convolutional neural network called 3D-ConvNet presented by [17] which learns spatio-temporal features for action recognition. Compared with 3D-ConvNet, our model is deeper and contains more feature maps.

Table 1. Average accuracy of the baseline method and the proposed 3D CNN approach

Method	Features	Ave Accuracy
GMM-HMM	Trajectory	81.1%
	Hand-shape	86.4%
	Trajectory + Hand-shape	90.8%
3D-ConvNet [17]	Gray channel	87.9%
3D CNN	Gray channel	88.5%
	Multi-channels	94.2%

4.3. Evaluation resluts

The average recognition accuracy rate of 3D CNN and GMM-HMM is shown in Table 1. We see that the average accuracy rate of GMM-HMM is 90.8% when using trajectory and hand-shape features, which has improvement compared with using trajectory or hand-shape features only. This result makes sense because for some sign actions, they are similar with respect to trajectory but the change of hand-shapes is huge (the same reason for hand-shapes only). Therefore, combining both features is a better choice. 3D CNN with only gray image as input achieves higher accuracy rate than 3D-ConvNet because our model use deeper architecture to generate more representative features. When using multiple chan-

nels data as input, the result outperforms all methods above. That is due to the fact that multi-channels of visual sources provide extra information for CNN and lead the model to paying attentions to changes in color, depth and trajectory instead of focusing on the gray channel only. 3D CNN has the capability to learn spatial-temporal features from raw video stream. Confusion matrix depicting the mis-classification is show in Fig. 6, which reveals most sign actions are recognized correctly.

5. CONCLUSION

We developed a 3D CNN model for sign language recognition. Our model learns and extracts both spatial and temporal features by performing 3D convolutions. The developed deep architecture extracts multiple types of information from adjacent input frames and then performs convolution and subsampling separately. The final feature representation combines information from all channels. We use multilayer perceptron classifier to classify these feature representations. For comparison, we evaluate both 3D CNN and GMM-HMM on the same dataset. The experimental results demonstrate the effectiveness of the proposed method.

6. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[2] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei, "Large-scale video classification with convolutional neural networks," in *CVPR*, 2014.

[3] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[4] Hueihan Jhuang, Thomas Serre, Lior Wolf, and Tomaso Poggio, "A biologically inspired system for action recognition," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*. Ieee, 2007, pp. 1–8.

[5] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu, "3D convolutional neural networks for human action recognition," *IEEE TPAMI*, vol. 35, no. 1, pp. 221–231, 2013.

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

[7] Thad Starner, Joshua Weaver, and Alex Pentland, "Real-time american sign language recognition using desk and wearable computer based video," *IEEE TPAMI*, vol. 20, no. 12, pp. 1371–1375, 1998.

[8] Christian Vogler and Dimitris Metaxas, "Parallel hidden markov models for american sign language recognition," in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. IEEE, 1999, vol. 1, pp. 116–122.

[9] Kouichi Murakami and Hitomi Taguchi, "Gesture recognition using recurrent neural networks," in *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 1991, pp. 237–242.

[10] Chung-Lin Huang and Wen-Yi Huang, "Sign language recognition using model-based tracking and a 3D hopfield neural network," *Machine vision and applications*, vol. 10, no. 5-6, pp. 292–307, 1998.

[11] Jong-Sung Kim, Won Jang, and Zeungnam Bien, "A dynamic gesture recognition system for the korean sign language (ksl)," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 26, no. 2, pp. 354–359, 1996.

[12] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *arXiv preprint arXiv:1311.2524*, 2013.

[13] Ronan Collobert and Jason Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *ICML*. ACM, 2008, pp. 160–167.

[14] Clément Farabet, Camille Couprie, Laurent Najman, and Yann LeCun, "Learning hierarchical features for scene labeling," *IEEE TPAMI*, vol. 35, no. 8, pp. 1915–1929, 2013.

[15] Srinivas C Turaga, Joseph F Murray, Viren Jain, Fabian Roth, Moritz Helmstaedter, Kevin Briggman, Winfried Denk, and H Sebastian Seung, "Convolutional networks can learn to generate affinity graphs for image segmentation," *Neural Computation*, vol. 22, no. 2, pp. 511–538, 2010.

[16] Ao Tang, Ke Lu, Yufei Wang, Jie Huang, and Houqiang Li, "A real-time hand posture recognition system using deep neural networks," *ACM Transactions on Intelligent Systems and Technology*, 2014.

[17] Moez Baccouche, Franck Mamalet, Christian Wolf, Christophe Garcia, and Atilla Baskurt, "Sequential deep learning for human action recognition," in *Human Behavior Understanding*, pp. 29–39. Springer, 2011.