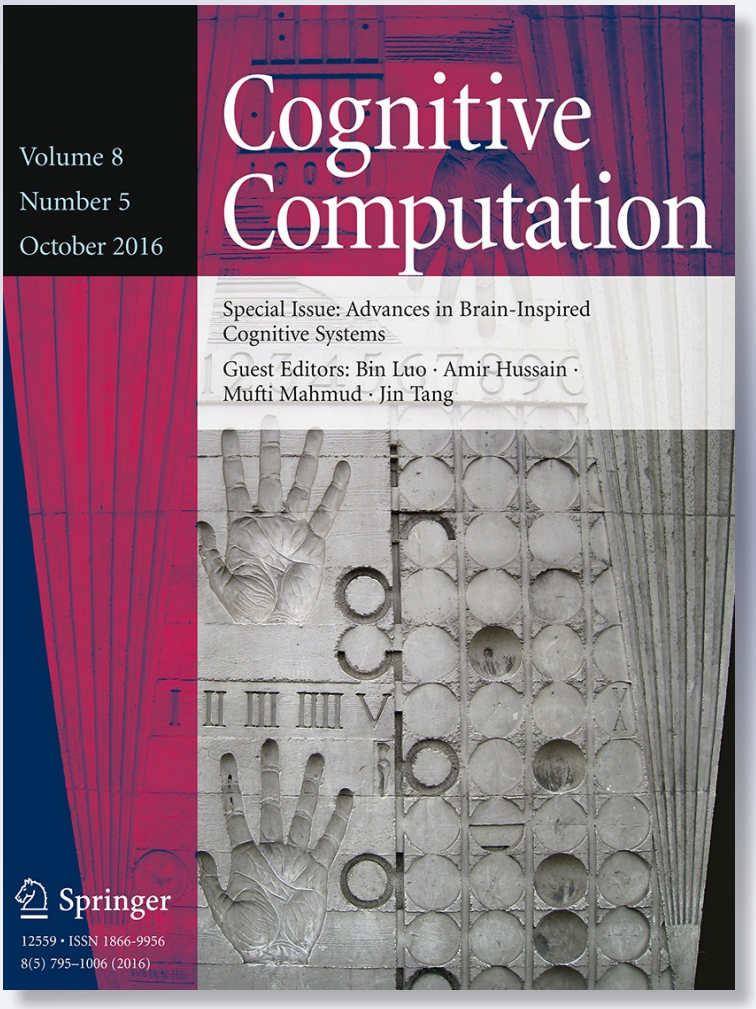
Sequentially Supervised Long Short-Term Memory for Gesture Recognition

Peisong Wang, Qiang Song, Hua Han & Jian Cheng

Cognitive Computation

ISSN 1866-9956
Volume 8
Number 5

Cogn Comput (2016) 8:982-991
DOI 10.1007/s12559-016-9388-6



Springer

Your article is protected by copyright and all rights are held exclusively by Springer Science +Business Media New York. This e-offprint is for personal use only and shall not be self-archived in electronic repositories. If you wish to self-archive your article, please use the accepted manuscript version for posting on your own website. You may further deposit the accepted manuscript version in any repository, provided it is only made publicly available 12 months after official publication or later and provided acknowledgement is given to the original source of publication and a link is inserted to the published article on Springer's website. The link must be accompanied by the following text: "The final publication is available at link.springer.com”.

Springer

Author's personal copy

Cogn Comput (2016) 8:982–991
DOI 10.1007/s12559-016-9388-6

Sequentially Supervised Long Short-Term Memory for Gesture Recognition

Peisong Wang[1] · Qiang Song[1] · Hua Han[1] · Jian Cheng[1]

Received: 2 October 2015 / Accepted: 19 February 2016 / Published online: 10 March 2016
© Springer Science+Business Media New York 2016

Abstract Gesture recognition has been suffering from long-term dependencies and complex variations in both spatial and temporal dimensions. Many traditional methods use hand cropping and sliding window scheme in the spatial and temporal space, respectively. In this paper, we propose a sequentially supervised long short-term memory architecture, which allows using pose information to guide the learning process of gesture recognition using variable length inputs. Technically, we add supervision at each frame using human joint positions. Our proposed methods can solve gesture recognition and pose estimation problems simultaneously using only RGB videos without hand cropping. Experimental results on two benchmark datasets demonstrate the effectiveness of the proposed framework compared with the state-of-the-art methods.

Keywords Gesture recognition · Pose estimation · LSTM · Sequential classification

✉ Jian Cheng
jcheng@nlpr.ia.ac.cn

Peisong Wang
peisong.wang@nlpr.ia.ac.cn

Qiang Song
qiang.song@nlpr.ia.ac.cn

Hua Han
hua.han@ia.ac.cn

[1] Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

Introduction

Gesture recognition has drawn increasing attention within the computer vision community. Developments have been driven by many applications such as human–computer interaction (HCI) [1, 2], robotics [3], sign language translation [4] as well as security and surveillance. However, effective gesture recognition is still a challenging task due to several factors such as different lighting conditions, cluttered backgrounds, motion blur, the small size of human hands in images, as well as different spatial and temporal scales. Traditional hand-designed features are inefficient to cope with such variations. For this reason, many traditional methods use a three stages recognition architecture [5–7] i.e., hand detection, hand tracking and recognition.

Deep and recurrent neural networks (DNNs and RNNs) are powerful learning models that achieve tremendous improvements on tasks such as image classification [8], object detection [9–11] and segmentation [12] as well as in the fields of natural language processing (NLP) and speech recognition [13]. Convolutional neural networks (CNNs) [14] have won numerous contests such as ILSVRC [8]. Recently, several works have adapted CNNs to the field of gesture recognition. Neverova et al. [15] propose a multi-scale and multi-modal deep architecture for gesture localization and recognition. They use fixed number of frames with different step sizes to solve different temporal scales. Pigou et al. [16] propose a MultiNet architecture based on CNN. They also use fixed frames and build separate CNN nets using full body images and cropped hand images. These methods outperform traditional gesture recognition methods. However, the results are not so satisfactory compared with the tremendous

Author's personal copy

improvements achieved by other tasks like image classification. There are several problems to think of:

1. *How to discover a feasible strategy to alleviate the underfitting and overfitting problems* [17]. While gestures have complex dependencies and dynamics in both spatial space and temporal space, deep neural networks tend to be much bigger for the task of gesture recognition than tasks like image recognition. What is more, gesture datasets are relatively smaller given that gesture labels are much harder to obtain than image labels. All these make the network hard to capture representative information which has good generalization.
2. *How to dispose of the dependency of hand positions* In order to get high classification accuracy, most current deep architectures for gesture recognition use image patches around human hands. Others may use 3D skeleton data. While in most real life applications, the human part positions are hard to obtain.
3. *How to distinguish temporal dimension from spatial dimensions* Most current deep architectures treat temporal dimension as another dimension of spatial space, which may destroy the characteristics of temporal space. Recurrent neural networks (RNNs) seem to be good architectures for temporal space; however, RNNs are mainly used for sequence labeling tasks. Using RNNs for classification of variable length sequences is still an open problem.

In this paper, we propose a **S**equentially **S**upervised **L**ong **S**hort-**T**erm **M**emory (SS-LSTM) architecture for gesture classification, which can partially solve the above-mentioned problems, i.e., the problem of underfitting/overfitting and the dependency of hand positions as well as treating temporal dimension separately. Instead of assigning class label to the output layer of RNNs, our SS-LSTM use auxiliary knowledge at every time step as *sequential supervision*. Our main inspiration is: pose estimation and gesture recognition are naturally correlated with each other. Specifically, gestures are different combinations of poses along temporal dimension. We use pose information as supervision to guide the learning process of gesture recognition. Our contributions lie in the following aspects:

1. We propose a Sequentially Supervised Long Short-Term Memory (SS-LSTM) model for classification of variable length inputs, which allows adding auxiliary knowledge to the learning process to alleviate underfitting and overfitting problems.
2. Our proposed methods can solve both pose estimation task and gesture recognition task simultaneously based on RGB data only, without any hand position acknowledgement.

Related Work

Traditional approaches for gesture recognition use hand-designed descriptors followed by classification. Works are mainly focused on the designing of local spatio-temporal descriptors and the study of different classification models. Recently many spatio-temporal descriptors are proposed, like the Harris3D [18], HOG3D [19] and Cuboids [20]. Wang et al. [21] evaluate different local descriptors at the same experimental settings. The authors of [22] extract HOG, HOF, MBHx, MBHy features along dense trajectories of gesture videos for classification and get state-of-the-art performance. One of the drawbacks of these hand-designed descriptors is the need of domain expertise. What is more, hand-designed descriptors are sensitive to variations, which makes it necessary to firstly detect body parts using object detection algorithms like [23, 24].

Classification models for gesture recognition are mainly of two kinds: sequential and non-sequential. Sequential approaches are mainly focused on how to model temporal dependencies of gestures. The work [25] combines the hidden Markov model (HMM) and (DTW) for gesture classification. In the work of [26], Wu and Cheng propose a Bayesian Co-Boosting method with Hidden Markov Mode for gesture recognition. Wu and Shao [27] propose to combine Deep Neural Networks with Hidden Markov Model to solve gesture segmentation and recognition problems simultaneously. Both these methods use multi-modal data for gesture recognition, especially depth data and 3D skeleton data of human body. Thanks to devices such as Kinect, these kinds of data are much easier to obtain than ever before. However, in most of our daily life, we have access to RGB data only, which hinders the usage of these kinds of methods.

Early deep learning methods train separate models for pose estimation [28, 29] and gesture recognition. Also there are approaches that use pose information to enhance the performance of the gesture classifier. These methods are mainly of two kinds. The first kind is to use articulated poses as features directly for classification [27]. The second case is to extract body part information given articulated poses. In the work of [15, 30], the authors use the cropped hand images as one modality to train a separate model to boost the recognition accuracy. All these methods require accurate tracking of body parts at *testing* time, which is a challenging task in its own right.

Author's personal copy

In contrast to all works described in this section, our approach combines human pose estimation and gesture recognition and learns spatial and temporal features directly from RGB videos only without hand cropping.

Proposed Method

In this section, we will first present our proposed SS-LSTM architecture followed by three different settings of how to use SS-LSTM for gesture recognition. To deal with different spatial variations, we use CNN network to extract spatial features of each frame. Then the CNN features are summed over by SS-LSTM network for temporal feature learning. The SS-LSTM network allows us to use variable length sequences as input.

Sequentially Supervised Long Short-Term Memory

Traditional RNNs can map input sequence $\mathbf{x} = (x_1, \ldots, x_t)$ to the hidden sequence $\mathbf{h} = (h_1, \ldots, h_t)$ and then the output sequence $\mathbf{y} = (y_1, \ldots, y_t)$ using the following equations:

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

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

Here the W terms denote weight matrices and b terms denote bias vectors. RNNs can be difficult to train using back-propagation algorithm, when there is long-term dependencies. This is often referred to as vanishing and exploding gradients problem [31].

These issues motivated the LSTM model which introduces a *memory cell* structure (Fig. 1, reproduced from [13]). A memory cell has a linear self-loop whose weights is 1.0 to force the *error flow* to be constant. The information exchanges between the memory cell and its environment are controlled by three sigmoid *gates*: the *input gate* i_t, *forget gate* f_t and *output gate* o_t. The carefully designed *memory cell* allows the LSTM model to learn long time intervals while maintaining short time lags. The updates of the LSTM units for time step t given inputs x_t, h_{t-1} and c_{t-1} are as follows:

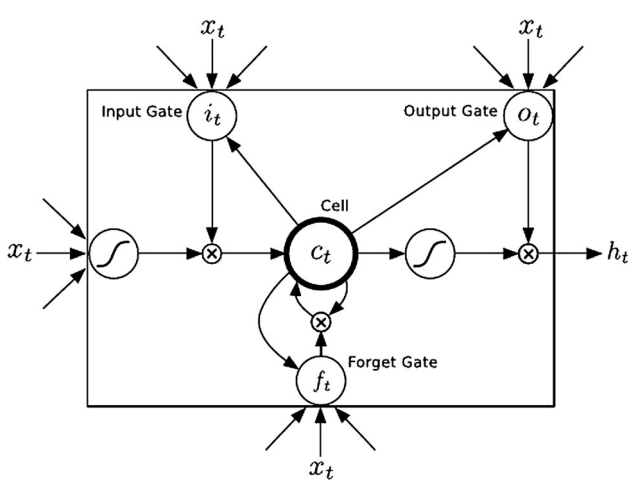


Fig. 1 Long short-term memory cell

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \quad (3)$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \quad (4)$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \quad (5)$$

$$g_t = \phi(W_{xc}x_t + W_{hc}ht - 1 + b_c) \quad (6)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (7)$$

$$h_t = o_t \odot \phi(c_t) \quad (8)$$

Here the $\sigma(x)$ is the *sigmoid* non-linear function and $\phi(x)$ is the *hyperbolic tangent* nonlinearity. The $\odot$ denotes an element-wise operator.

Recurrent neural networks, including LSTM, are successfully used in time series prediction problems, such as machine translation, natural language process and music composition. In all these tasks, the outputs of RNNs are time series. But for classification, the output is a constant class label. There are two commonly used strategies to utilize RNNs for classification. We refer these two strategies as *Lastly Supervised LSTM* (LS-LSTM, Fig. 2a) and *Deeply Supervised LSTM*(DS-LSTM, Fig. 2b). Here v_t indicates input vector at time step t and y indicates the class label of current input sequence. The supervision of the LS-LSTM method is at the final LSTM cell. The effect of the supervision gets weaker for the earlier time steps. In order to learn meaningful features, we need more training data and more training iterations. On the other hand, the DS-LSTM "pushes too hard" on the classifier. It demands the classifier to assign the right class label without seeing the whole sequence. This will introduce inaccurate supervising information for the earlier feature learning stages of the RNNs. For example, considering a toy example of two class gesture classification task: Class-A, move right and then go up (Fig. 3a), and Class-B, move right and then go down (Fig. 3b). The earlier stages of these two classes are almost the same and the only difference is in the later stages. If we assign different labels for the similar earlier stages of sequences from the two classes, the classifier will learn the tiny difference of the details which is noise for distinguish the two classes.

Our proposed SS-LSTM, on the other hand, can assign "right label" to every time step in the sequence (Fig. 2c). Here the "right label" is not the class label, but the human knowledge for the classification learning process. Considering the toy example again, we assign "moving right" label for the earlier stages of both classes, while assign "moving up" label for later stages of Class-A and "moving down" label for later stages of Class-B. More specifically,

Author's personal copy

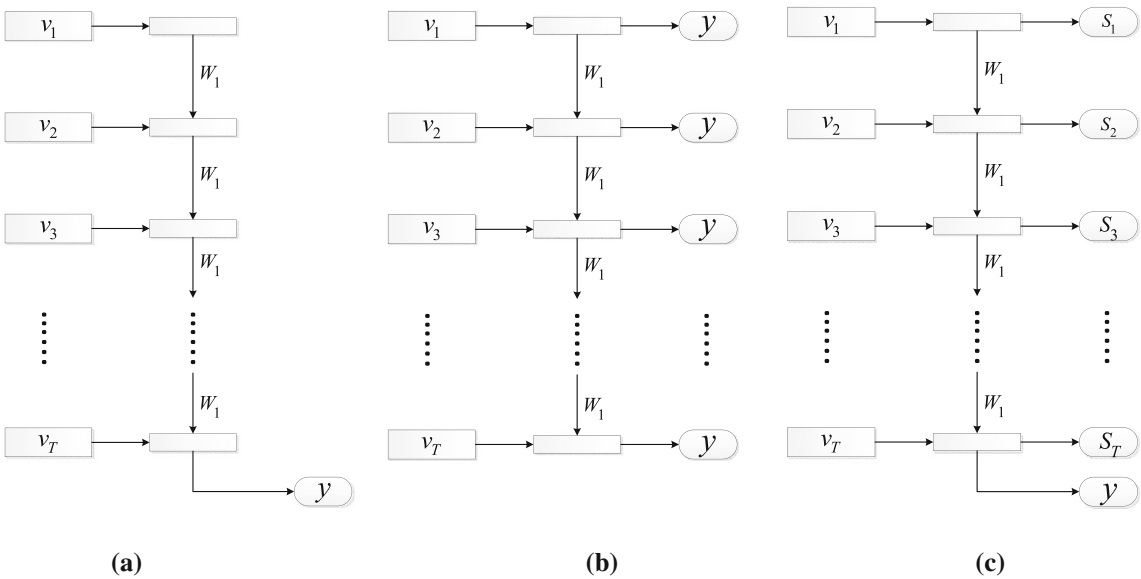


Fig. 2 LSTM classification model. **a** LS-LSTM, **b** DS-LSTM and **c** SS-LSTM

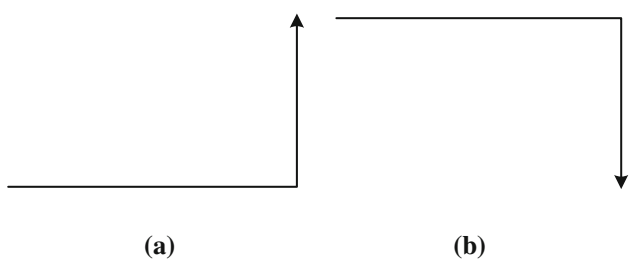


Fig. 3 A toy example of gesture classification. **a** Class-A and **b** Class-B

for our gesture classification task, we choose the human joint positions as the "right label".

Sequentially Supervised LSTM for Gesture Recognition

Our proposed SS-LSTM for gesture recognition model includes three different settings: (1) pose pre-trained feature learning for gesture recognition; (2) encoder–decoder SS-LSTM for gesture recognition and (3) multi-task learning for gesture recognition and pose estimation. The whole architecture includes four blocks: the input block, the CNN block, the LSTM block, and the output block (see Fig. 4). The input block includes sequences of different lengths. The CNN block is used to extract spatial appearance features of each frame. Our SS-LSTM architecture corresponds to the last two blocks, i.e., the LSTM block and the output block. We utilize human joint positions as the sequential supervision for our SS-LSTM. The difference between these three settings lies in the output blocks.

SS-LSTM-1: Pose Pre-trained Feature Learning

An intuitive approach to incorporate pose information for gesture recognition is to use the pre-training and fine-tuning scheme. The intuition behind this method is that gestures are combinations of different poses, so we can transfer knowledge learnt from pose estimation to gesture recognition. Based on this view, the parameters learnt from pose estimation task should be good initializations for the task of gesture recognition.

We first train the network as a *pose regression* problem. More precisely, given a sequence of frames, we pass it through the CNN block and then the LSTM layers and output the estimated locations of the human upper-body joints at each time step. We employ the l_2 loss function, which penalizes the l_2 distance between the pose predictions and the ground truth. Note that we normalize the locations of human joints to the range of [0, 1].

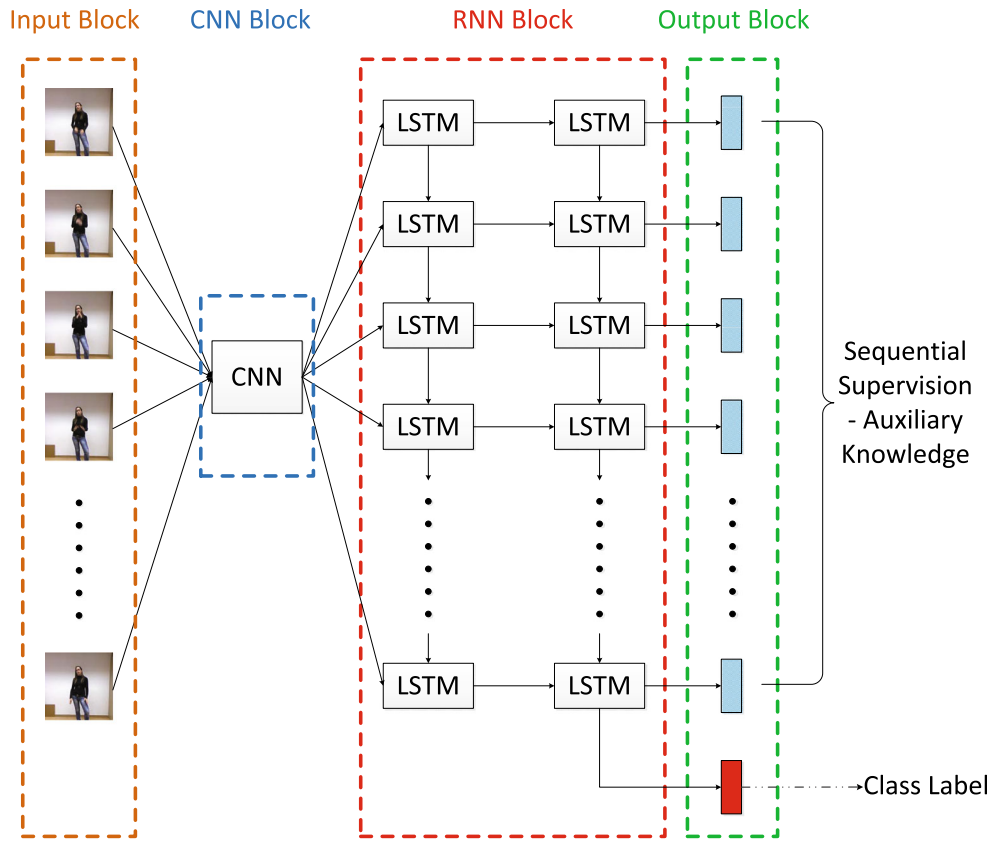
We denote (x, y, J) as a training sample, where y denotes gesture label and J stands for the vector of normalized coordinates of k joints in the image x. Given a training set $M = \{(x, y, J)\}$, the training objective is to estimate the network parameters by minimizing the loss function:

$$\mathcal{L} = \sum_{m=1}^{M} \sum_{t=1}^{T} \| \hat{J}_t^m - J_t^m \|_2^2 \tag{9}$$

Once the pose regression network is learnt, we could then use it either as an initialization or a fixed feature extractor for the task of gesture recognition. In the later case, we will remove the last fully connected layer as well as the l_2 loss layer, then treat the rest of the network as a fixed feature extractor to obtain the last LSTM output at the

Author's personal copy

Fig. 4 Our CNN-LSTM model



last time step T. Here, we adopt the former approach, i.e., to fine-tune the gesture classifier with the learnt parameters.

There are several advantages of this setting. First, using pose information to supervise the learning process can achieve boosted recognition accuracy by avoiding underfitting and overfitting. Second, this method allows using datasets which have only joint positions but not gesture labels. Thanks to devices such as Kinect, this kind of samples are much easier to obtain because there is no need for tedious human labelling work.

SS-LSTM-2: Encoder–Decoder Feature Learning

In this section, we describe the encoder–decoder framework to learn efficient representations for gesture recognition. The LSTM encoder–decoder frameworks obtain tremendous improvements in machine translation. The task of machine translation is to map a sentence written in a source language to a sentence of the target language. Our SS-LSTM encoder–decoder method can be seen as a special kind of machine translation from the view that, both the *frame flow* and *pose flow* can be seen as a "language" describing the specific gesture.

The encoder LSTM takes in the input frames and generates the representation which is the output of the last LSTM cell at time step T. The decoder LSTM, on the other hand, takes the representation as input and outputs the user's joint positions of each frame. Note that the output of the decoder LSTM is in reverse order of the input frames. What is more, the decoder LSTM cells receive the last generated output joint positions as input, i.e., the dotted line shown in Fig. 5. The benefit is to make the learning process much easier. In this way, we force the network to remember the configurations of human body in each frame while discarding information that is irrelevant to the task of gesture classification such as the background and the different body sizes.

SS-LSTM-3: Multi-task Feature Learning

Gesture recognition and pose estimation are naturally correlated with each other. Different gestures are the different combinations of poses in the temporal space. However, existing deep learning models either treat them separately or combine them in a loose way. In this section, we propose to train a single network for the two tasks simultaneously. The network architecture is shown in Fig. 6.

The loss function for the network is of two parts, the *Regression Loss* $\mathcal{L}_r$ and the *Classification Loss* $\mathcal{L}_c$.

Author's personal copy

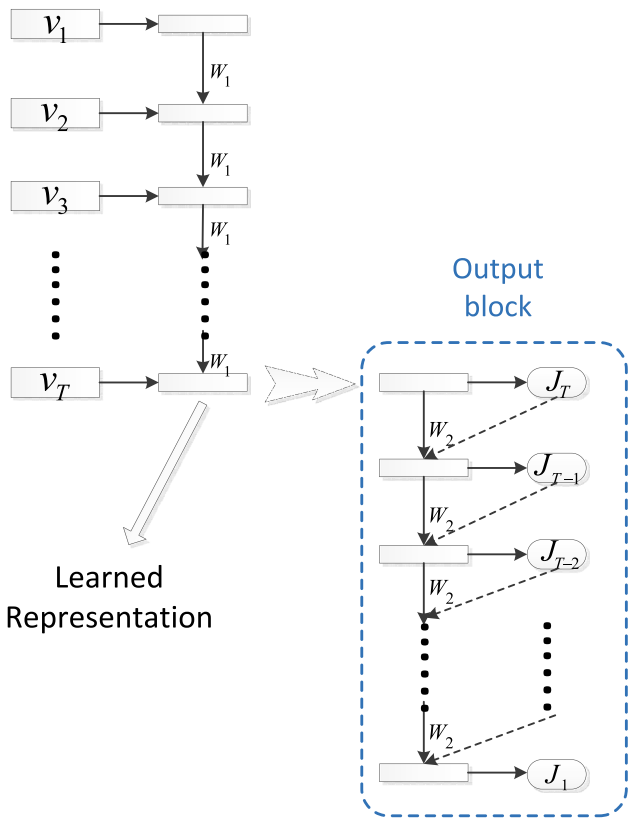


Fig. 5 Pose encoder–decoder feature learning: output block

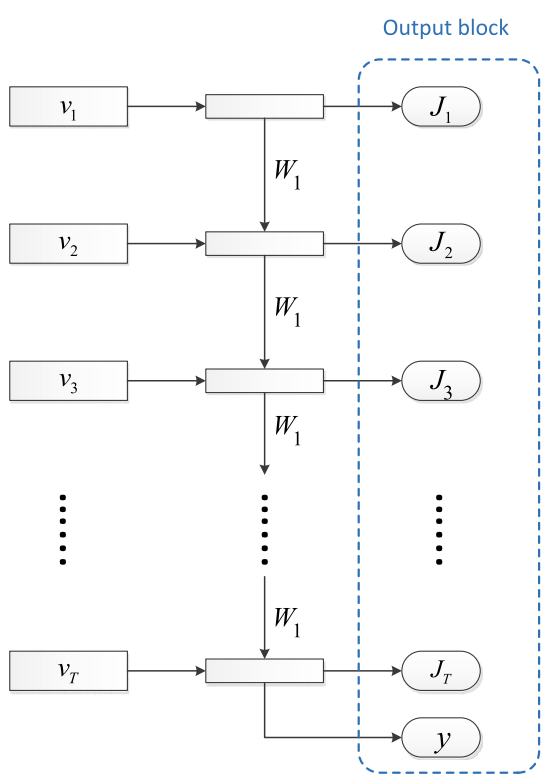


Fig. 6 Multi-task feature learning: output block

$$\mathcal{L}_r = \sum_{m=1}^{M} \sum_{t=1}^{T_m} \| \hat{J}_t^m - J_t^m \|_2^2 \tag{10}$$

$$\mathcal{L}_c = \sum_{m=1}^{M} \frac{-\exp\left(y^m[k]\right)}{\sum_{k'=1}^{K} \exp\left(y^m[k']\right)} \tag{11}$$

$$\mathcal{L} = \alpha \mathcal{L}_r + \beta \mathcal{L}_c \tag{12}$$

Our proposed SS-LSTM model allows us to use pose information at every time step as supervision. By incorporating "the right" supervision at every time step, as well as the ultimate supervision at the last time step, the model can learn relevant spatial–temporal features for the task of gesture recognition. Note that although our model is capable of learning two tasks simultaneously, our final purpose is to enhance the classification ability. So the pose regression task can also be seen as a regularization term for the gesture classification task.

Experiments and Results

To evaluate the effectiveness of avoiding underfitting and overfitting problems of our proposed SS-LSTM framework, we conduct experiments on a relatively smaller dataset, i.e., the ChaAirGest dataset [32]. We also conduct experiments on the Chalearn LAP2014 dataset [33], which is among the largest public dataset for gesture recognition. Comparisons between our methods and current state-of-the-art methods demonstrate the effectiveness of our proposed methods.

Datasets

ChAirGest [32]

The ChAirGest dataset contains 1200 gesture instances from 10 different gestures. The gestures are recorded in two different lighting conditions by 10 different subjects. The dataset is captured with a Kinect camera and 4 inertial motion units (IMUs) attached to the right arm and the neck of the subject. In our experiments we use only the RGB data of the Kinect camera, which makes it more challenging than using IMU data and skeleton data. To avoid seeing the same subject at both training and testing stages, we randomly select 8 of the 10 subjects for training and the other 2 subjects for testing.

LAP2014 [33]

The Chalearn LAP2014 dataset is composed of total 940 sequences (470 training, 230 validation, and 240 test sequences). Specifically, there are total 13,856 instances of Italian conversational gestures performed by different

Author's personal copy

people and recorded with a consumer RGB-D sensor. It includes color, depth video, articulated pose streams, and manually annotated gesture labels. The gestures are drawn from 20 categories of Italian sign gestures. The lengths of gestures range from 16 frames to 100 frames. This is one of the largest public datasets for gesture recognition.

Implementation Details

Network Architecture

The spatial network is used to extract static appearance features from each frame. In our implementation, we choose the AlexNet [8] architecture. Note that we only need the five convolutional layers and discard the fully connection layers as well as the softmax layer. The temporal network is composed of two LSTM layers, each has 1024 hidden units. For final classification, the last LSTM unit (with time step T) of the last LSTM layer is connected to a softmax layer.

Training

All gestures are resized to have a frame size of 256×256. On training, we randomly crop input videos into the size of 227×227. For the dataset of LAP2014, we also horizontally flip them with 50 % probability. We do not flip the ChAirGest dataset because all subjects use their left hand. For temporal consistency, we do the same crop or mirror for all frames of a video. We use SGD with momentum 0.9 for training our network. With current GPU memory, we choose the batch size of 10. To make all sequences within a batch the same length, we pad the shorter ones with empty frame (which have all values of zero). For each sequence, we use a binary vector of the same length to indicate whether the frames are valid or empty frame. Our implementation is under the framework of Theano [34, 35].

Comparison of Different Models

In this section, we will compare the different settings of our proposed method and the current state-of-the-art methods.

Our Three SS-LSTM Settings

(1) Pose pre-trained SS-LSTM for gesture recognition; (2) encoder–decoder SS-LSTM for gesture recognition and (3) multi-task SS-LSTM for gesture recognition and pose estimation.

In all of our experiments, we use whole RGB sequences only without hand cropping for the final gesture classification.

HMM-Based Methods

We compare our methods with HMM-based methods to show that our methods can model temporal information more efficiently. Here we use the results published in the work of [36].

LRCN Color

We also choose the LRCN [37] model, which is very like to our SS-LSTM model, as the baseline to see if the SS-LSTM model can help improve performance on gesture recognition task. We use the same setting of their released model. The input video is cropped to a fixed length of 16 frames. To train the LRCN model, they firstly train the spatial net (the CNN layers) using each frames which share the same label with the gesture video. They also pre-train the spatial net using ILSVRC-2012 [38] classification training subset of the ImageNet dataset to prevent overfitting. After the spatial net is trained, they fine-tune the whole net end-to-end.

CNN MultiNet

Many previous deep learning for gesture recognition approaches use sliding window method and crop out human body parts for classification. The winner [15] of Chalearn LAP2014 challenge also use this method, which they call "path", as part of their network for the final fusion. But they did not published their classification result. Here we use the results published in the work of [16]. They train two CNNs, one for extracting hand features and one for extracting upper-body features. Then they combine the results together for the final classification.

Results

Results on ChAirGest

The comparison results of our proposed methods and the baseline method on ChAirGest dataset are shown in Table 1.

Table 1 Comparison results on ChAirGest

Model	Gesture accuracy	Pose error	Modalities
LRCN color [37]	69.7	–	RGB
SS-LSTM-1	90.6	0.026	RGB
SS-LSTM-2	91.8	0.032	RGB
SS-LSTM-3	93.9	0.037	RGB

Author's personal copy

Table 2 Comparison results on LAP2014

Model	Gesture accuracy	Pose error	Modalities
HMM color [36]	59.1	–	RGB
HMM fusion [36]	72.6	–	RGB, depth, skeleton
CHMM fusion [36]	73.5	–	RGB, depth, skeleton
CNN MultiNet [16]	91.7	–	Gray, depth, body image, cropped hand
LRCN color [37]	87.8	–	RGB
SS-LSTM-1	88.5	0.010	RGB
SS-LSTM-2	92.0	0.018	RGB
SS-LSTM-3	93.9	0.029	RGB

From the results we can see that all our methods outperform the LRCN color baseline method. During our experiments, we notice that the LRCN color model overfits easily on the training set even with high dropout. By contrast, all our methods can learn effective representations without dropout, which shows the effectiveness of preventing overfitting.

Results on LAP2014

Table 2 illustrates the experimental results on LAP2014 dataset. Our method can solve the pose estimation task at the same time. So we also give the normalized pose regression error, which is the mean pixel error divided by image width or height. It is clear that our proposed method outperforms the baseline LRCN model and the CNN MultiNet model which combines the nets trained using upper-body image and the cropped hand image separately.

The results of our three settings are reasonable. We will give a detailed discussion below.

Benefiting from the sequential supervision of the pose information at every frame, all our methods outperform the baseline method. The more intuitive pose pre-training method incorporates pose information but cannot manage to learn temporal dependencies required by gesture classification. Because we can predict the joint positions just from the current frame. We can get the same conclusion from the pose estimation accuracy of our methods. Our pose pre-training method learns too much information about the joint positions but does not care about the final task of gesture recognition. In other words, it overfits for the pose estimation task, resulting bad generalization for gesture classification.

The encoder–decoder method gets better result for the task of gesture recognition. We argue that the encoder–decoder method forces the network to remember all the joint positions. This helps a lot for gesture classification. The problem here is that the pose guidance is only at the decoder part of the network. This can be hard for the training of the encoder LSTM.

Our multi-task method obtains best result of all methods. This method combines pose guidance with the final gesture supervision. The pose estimation task encourages the net to learn along the "right direction", while the gesture recognition task makes the net to learn only relevant features and discard noise for the task of gesture recognition.

Conclusions and Summary

In this paper, we propose a SS-LSTM method for gesture recognition. We use the pose information as the guidance for the learning process, which makes the network to capture representative information that generalize well. Note that the pose information is only used during the training stage. The benefit is that we can use samples with only pose information but no gesture labels. This kind of samples are easy to get thanks to devices like Kinect. We only use RGB data for final gesture recognition. This is important because most of actual life applications have only RGB cameras. Our method can also solve pose estimation task at the same time. This is another contribution of our work.

Acknowledgments This work was supported in part by National Natural Science Foundation of China (Grant No. 61332016) and Project of Chinese Academy of Sciences (Grant No. XDB02060001).

Compliance with Ethical Standards

Conflict of Interest Peisong Wang, Qiang Song, Hua Han and Jian Cheng declare that they have no conflict of interest.

Informed Consent All procedures followed were in accordance with the ethical standards of the responsible committee on human experimentation (institutional and national) and with the Helsinki Declaration of 1975, as revised in 2008 (5). Additional informed consent was obtained from all patients for which identifying information is included in this article.

Human and Animal Rights This article does not contain any studies with human participants performed by any of the authors.

Author's personal copy

References

1. Rautaray SS, Agrawal A. Adaptive hand gesture recognition system for multiple applications. In: Agrawal A, Tripathi RC, Yi-Luen Do E, Tiwari MD, editors. Intelligent interactive technologies and multimedia. Berlin: Springer; 2013. p. 53–65.
2. Squartini S, Schuller B, Hussain A. Cognitive and emotional information processing for human–machine interaction. Cogn Comput. 2012;4(4):383–5.
3. Xu D, Wu X, Chen YL, Xu Y. Online dynamic gesture recognition for human–robot interaction. J Intell Robot Syst. 2014;77(3–4):583–96.
4. Kröger BJ, Birkholz P, Kannampuzha J, Kaufmann E, Mittelberg I. Movements and holds in fluent sentence production of American sign language: the action-based approach. Cogn Comput. 2011;3(3):449–65.
5. Rautaray SS, Agrawal A. Vision based hand gesture recognition for human–computer interaction: a survey. Artif Intell Rev. 2015;43(1):1–54.
6. Shi MY, Zhan DC. Multi gesture recognition: a tracking learning detection approach. In: Sun C, Fang F, Zhou Z-H, Yang W, Liu Z-Y, editors. Intelligence science and big data engineering. Berlin: Springer; 2013. p. 714–21.
7. Fang Y, Wang K, Cheng J, Lu H. A real-time hand gesture recognition method. In: 2007 IEEE international conference on multimedia and expo. USA: IEEE; 2007. p. 995–98
8. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems (NIPS). 2012. p. 1106–14
9. Girshick RB, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE conference on computer vision and pattern recognition (CVPR). 2014. p. 580–7
10. Girshick RB. Fast R-CNN. CoRR abs/1504.08083 (2015)
11. Ren S, He K, Girshick RB, Sun J. Faster R-CNN: towards real-time object detection with region proposal networks. CoRR abs/1506.01497 (2015)
12. Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. CoRR abs/1411.4038 (2014)
13. Graves A, Mohamed Ar, Hinton G. Speech recognition with deep recurrent neural networks. In: 2013 IEEE international conference on acoustics, speech and signal processing (ICASSP). USA: IEEE; 2013. p. 6645–9.
14. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. In: Haykin S, Kosko B, editors. Intelligent signal processing. USA: IEEE; 2001. p. 306–51.
15. Neverova N, Wolf C, Taylor GW, Nebout F. Multi-scale deep learning for gesture detection and localization. In: Computer vision-ECCV 2014 workshops. Berlin: Springer; 2014. p. 474–90.
16. Pigou L, Dieleman S, Kindermans PJ, Schrauwen B. Sign language recognition using convolutional neural networks. In: Computer vision-ECCV 2014 workshops. Berlin: Springer; 2014. p. 572–8.
17. Sutskever I, Martens J, Dahl G, Hinton G. On the importance of initialization and momentum in deep learning. In: Proceedings of the 30th international conference on machine learning (ICML-13); 2013. p. 1139–47.
18. Sipiran I, Bustos B. Harris 3D: a robust extension of the harris operator for interest point detection on 3d meshes. Vis Comput. 2011;27(11):963–76.
19. Klaser A, Marszałek M, Schmid C. A spatio-temporal descriptor based on 3d-gradients. In: BMVC 2008—19th British Machine Vision Conference. British Machine Vision Association; 2008. p. 275–1.
20. Dollár P, Rabaud V, Cottrell G, Belongie S. Behavior recognition via sparse spatio-temporal features. In: 2nd joint IEEE international workshop on visual surveillance and performance evaluation of tracking and surveillance, 2005. USA: IEEE; 2005. p. 65–72.
21. Wang H, Ullah MM, Klaser A, Laptev I, Schmid C. Evaluation of local spatio-temporal features for action recognition. In: BMVC 2009—British Machine Vision Conference. BMVA Press; 2009. p. 124–31.
22. Peng X, Wang L, Cai Z, Qiao Y. Action and gesture temporal spotting with super vector representation. In: Computer vision-ECCV 2014 workshops. Berlin: Springer; 2014. p. 518–27.
23. Zhang H, Bai X, Zhou J, Cheng J, Zhao H. Object detection via structural feature selection and shape model. IEEE Trans Image Process. 2013;22(12):4984–95.
24. Tu Z, Zheng A, Yang E, Luo B, Hussain A. A biologically inspired vision-based approach for detecting multiple moving objects in complex outdoor scenes. Cogn Comput. 2015;7(5):539–51.
25. Wu J, Cheng J, Zhao C, Lu H. Fusing multi-modal features for gesture recognition. In: Proceedings of the 15th ACM on international conference on multimodal interaction. New York: ACM; 2013. p. 453–60.
26. Wu J, Cheng J. Bayesian co-boosting for multi-modal gesture recognition. J Mach Learn Res. 2014;15(1):3013–36.
27. Wu D, Shao L. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. In: 2014 IEEE conference on computer vision and pattern recognition (CVPR). USA: IEEE; 2014. p. 724–31.
28. Toshev A, Szegedy C. Deeppose: Human pose estimation via deep neural networks. In: 2014 IEEE conference on computer vision and pattern recognition (CVPR). USA: IEEE; 2014. p. 1653–60.
29. Tompson JJ, Jain A, LeCun Y, Bregler C. Joint training of a convolutional network and a graphical model for human pose estimation. In: Advances in neural information processing systems; 2014. p. 1799–807.
30. Neverova N, Wolf C, Paci G, Sommavilla G, Taylor GW, Nebout F. A multi-scale approach to gesture detection and recognition. In: 2013 IEEE international conference on computer vision workshops (ICCVW). USA: IEEE; 2013. p. 484–91.
31. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80.
32. Ruffieux S, Lalanne D, Mugellini E. Chairgest: a challenge for multimodal mid-air gesture recognition for close HCI. In: Proceedings of the 15th ACM on international conference on multimodal interaction. USA: ACM; 2013. p. 483–88.
33. Escalera S, Baró X, Gonzalez J, Bautista MA, Madadi M, Reyes M, Ponce-López V, Escalante HJ, Shotton J, Guyon I. Chalearn looking at people challenge 2014: dataset and results. In: Computer vision-ECCV 2014 workshops. Berlin: Springer; 2014. p. 459–73.
34. Bergstra J, Breuleux O, Bastien F, Lamblin P, Pascanu R, Desjardins G, Turian J, Warde-Farley D, Bengio Y. Theano: a CPU and GPU math expression compiler. In: Proceedings of the Python for scientific computing conference (SciPy), vol. 4. Austin, TX; 2010. p. 3.
35. Bastien F, Lamblin P, Pascanu R, Bergstra J, Goodfellow I, Bergeron A, Bouchard N, Warde-Farley D, Bengio Y. Theano: new features and speed improvements. arXiv preprint arXiv:1211.5590 (2012).
36. Cao C, Zhang Y, Lu H. Multi-modal learning for gesture recognition. In: 2015 IEEE international conference on multimedia and expo (ICME). USA: IEEE; 2015. p. 1–6.

Author's personal copy

37. Donahue J, Hendricks LA, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T. Long-term recurrent convolutional networks for visual recognition and description. arXiv preprint arXiv:1411.4389 (2014).

38. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, et al. Imagenet large scale visual recognition challenge. Int J Comput Vis. 2015;115(3):211–52.