Multimodal Gesture Recognition using Multi-stream Recurrent Neural Network

Noriki Nishida, Hideki Nakayama

Machine Perception Group
Graduate School of Information Science and Technology
The University of Tokyo
nishida@nlab.ci.i.u-tokyo.ac.jp
nakayama@ci.i.u-tokyo.ac.jp

Abstract. In this paper, we present a novel method for multimodal gesture recognition based on neural networks. Our multi-stream recurrent neural network (MRNN) is a completely data-driven model that can be trained from end to end without domain-specific hand engineering. The MRNN extends recurrent neural networks with Long Short-Term Memory cells (LSTM-RNNs) that facilitate the handling of variable-length gestures. We propose a recurrent approach for fusing multiple temporal modalities using multiple streams of LSTM-RNNs. In addition, we propose alternative fusion architectures and empirically evaluate the performance and robustness of these fusion strategies. Experimental results demonstrate that the proposed MRNN outperforms other state-of-the-art methods in the Sheffield Kinect Gesture (SKIG) dataset, and has significantly high robustness to noisy inputs.

Keywords: multimodal gesture recognition, recurrent neural networks, long short-term memory, convolutional neural networks

1 Introduction

Deep neural networks are efficient machine learning models used by many applications in computer vision, speech recognition, and natural language processing. Although various architectures have been proposed in recent years, convolutional neural networks (ConvNets) are currently dominant in a variety of benchmarks in computer vision [1, 2]. In many object recognition competitions, such as the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), nearly all top-ranked teams used ConvNets. In recent studies, the error rate of the state-of-the-art models outperforms the human performance [3, 4].

On the other hand, in gesture recognition, such a dominant model has not appeared yet. One main reason is that it is difficult for neural networks to simultaneously learn effective image representations and sequential models.

Traditional gesture recognition systems consist of several consecutive stages [5,

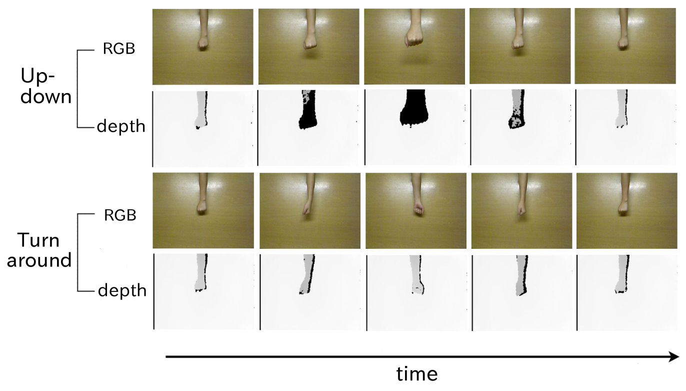


Fig. 1. Two examples in the SKIG dataset [10]. This figure shows that each modality has different importance for different gestures. The upper two rows show a color sequence and a corresponding depth sequence of an example from the "Up-down" category, whereas the lower two rows show an example from the "Turn around" category. It can be seen that depth modality is useful for classifying the upper example. However, in the lower example, the temporal change in the depth modality is very minor, because the key action of the "Turn around" gesture is a hand rotation. Therefore, the color modality should also be considered in classification.

6]. The first stage involves detecting and segmenting the regions of the objects being focused on (e.g., hands, arms). This stage requires prior knowledge of target gesture domains. In the second stage, features are extracted from the segmented regions. Finally, the last stage classifies input gestures using sequential data models such as the Hidden Markov Model (HMM). One of the largest drawbacks of this consecutive approach is that overall performance strongly depends on the quality and the generalization ability of each stage. Moreover, hand-coded heuristics (such as skin color filtering) that are often used during the first stage make the entire system too specific to the target gestures.

Hand-coded heuristics lead to a lack of generality in gesture recognition systems. Representation learning is one of the most efficient methods for addressing this problem [7]. Representation learning focuses on the extraction of effective features from raw data. In particular, multimodal representation learning has attracted increasing attention in machine learning [8, 9]. One main reason for this trend is that multimodal information can improve the robustness of classifiers. In general, an object can be described by various modalities, including color image, depth data, sound, and natural language. Each modality is expected to carry different information. Some objects might not be correctly discriminated from others if only a single modality is available. Information from multiple modalities can suppress this type of misclassification. This is also the case for gesture recognition. Some gestures are similar in color modality but are significantly different in depth modality, and vice versa. We show such examples from the SKIG dataset [10] in Fig. 1. The upper two rows are a color sequence and a corresponding depth sequence of an example video from the "Up-down" category. The lower two rows show another example from the "Turn around" category. It is obvious that the depth information is effective for classifying the "Up-down" gesture. However, the "Turn around" gesture requires color information, because the depth modality changes very slightly throughout the frames. Thus, incorporating multimodal information is crucial for improving classification performance.

This paper makes the following contributions:

- We introduce a novel multimodal-temporal fusion approach that uses recurrent neural networks with Long Short-Term Memory cells (LSTM-RNNs) [11, 12]. Our recurrent fusion method makes it possible to embed multiple modalities while considering temporal dynamics.
- We also propose alternative architectures as baselines to fuse multiple temporal modalities and compare them with the MRNN.
- We show for the first time that fusing multiple modalities while considering temporal dynamics is significantly beneficial not only for improving classification performance, but also for increasing robustness to noisy inputs.
- The MRNN outperforms previous approaches and our alternative models in the SKIG dataset, and achieves state-of-the-art performance.

2 Related work

Our work continues in the path established by Murakami et al. (1991) [13]. Both their work and ours propose a data-driven method for gesture recognition that does not require any prior knowledge of target gesture domains. Gesture recognition systems that use a more conventional approach split the entire system into three components: a hand detector, feature extractor, and classifier [5, 6]. While there are many works that employ machine learning techniques for gesture recognition, detection and segmentation of key objects (e.g., hands, arms) remain hand-coded. In contrast, in our MRNN, these processes are automatically optimized towards end-to-end classification performance.

Many neural network models have been applied to various tasks in computer vision, including object recognition [14], [3], object detection [2], semantic segmentation [15], and image generation [16]. Murakami et al. (1991) [13] used Elmen RNN [11] for gesture recognition. They used data collected from data gloves. However, in practice, data gloves are not always available for real-world applications. Ji et al. (2013) [17] extended conventional ConvNets to 3D ConvNets for handling videos. Karpathy et al. (2014) [18] proposed hierarchically stacked 2D ConvNets for fusing temporal-spatial information. Donahue et al. (2014) [19] incorporated a ConvNet with an LSTM-RNN and applied it to action recognition. Molchanov et al. (2015) [20] proposed a multimodal gesture recognition model consisting of two streams of 3D ConvNets. They developed a high-resolution 3D ConvNet and a low-resolution 3D ConvNet that are merged in the last layer, and achieved the highest accuracy in the 2015 Vision for Intelligent Vehicles and Applications (VIVA) challenge [9]. Their approach is similar to our early multimodal fusion model; however, we use LSTM-RNNs to extract temporal dynamics. Liu et al. (2013) [10] proposed restricted graph-based genetic programming to fuse color and depth modalities. However, their approach requires primitive 3D operations that must be defined before training. The process of choosing these operations limits the model's classification performance. In this paper, we use multiple streams of LSTM-RNNs to fuse multiple temporal modalities. The recurrent nature of our approach allows it to fuse modalities sequentially while considering temporal dependencies. For comparison, we propose alternative architectures that fuse modalities before or after LSTM-RNN streams, and demonstrate that the proposed MRNN is a significantly efficient model in terms of classification performance and robustness to noisy inputs.

3 Multi-stream Recurrent Neural Network

Overview The purpose of our model is to classify gestures into given categories by utilizing information from multiple modalities. We develop multiple streams of LSTM-RNNs using ConvNets. Each stream receives frame-level inputs at every step from corresponding modalities, and independently represents the temporal dynamics of each modality. To embed the disconnected representation into a common space, we construct an additional LSTM-RNN stream on top of these streams. Fig. 2 displays the graphical representation of the MRNN. For comparison, we also propose two alternative architectures: the late multimodal fusion model and the early multimodal fusion model. These models are displayed in Fig. 3 (a), (b). In this section, we first explain LSTM-RNNs. We then describe the details of our proposed models.

3.1 RNN with LSTM cells

A recurrent neural network (RNN) is a straightforward extension of multilayer perceptrons to sequential modeling [11]. Let $\boldsymbol{\phi}(\mathbf{x}_t) \in \mathbb{R}^n$ and $\mathbf{h}_t \in \mathbb{R}^m$ be a nonlinearly transformed input and a hidden state, respectively, at step t. The nonlinear function $\boldsymbol{\phi}$ is a neural network that extracts the feature vector from the input $\mathbf{x}_t$. For example, ConvNets can be used as $\boldsymbol{\phi}$ if $\mathbf{x}_t$ is spatial data such as a color image. The neural network $\boldsymbol{\phi}$ is a portion of the entire architecture, and is optimized simultaneously with the remaining network. Given input sequence $(\boldsymbol{\phi}(\mathbf{x}_1), \ldots, \boldsymbol{\phi}(\mathbf{x}_T))$, an RNN computes the hidden sequence $(\mathbf{h}_1, \ldots, \mathbf{h}_T)$ using the following equations:

$$\mathbf{h}_t = f(\mathbf{W}_{in}\boldsymbol{\phi}(\mathbf{x}_t) + \mathbf{W}_{hh}\mathbf{h}_{t-1}). \tag{1}$$

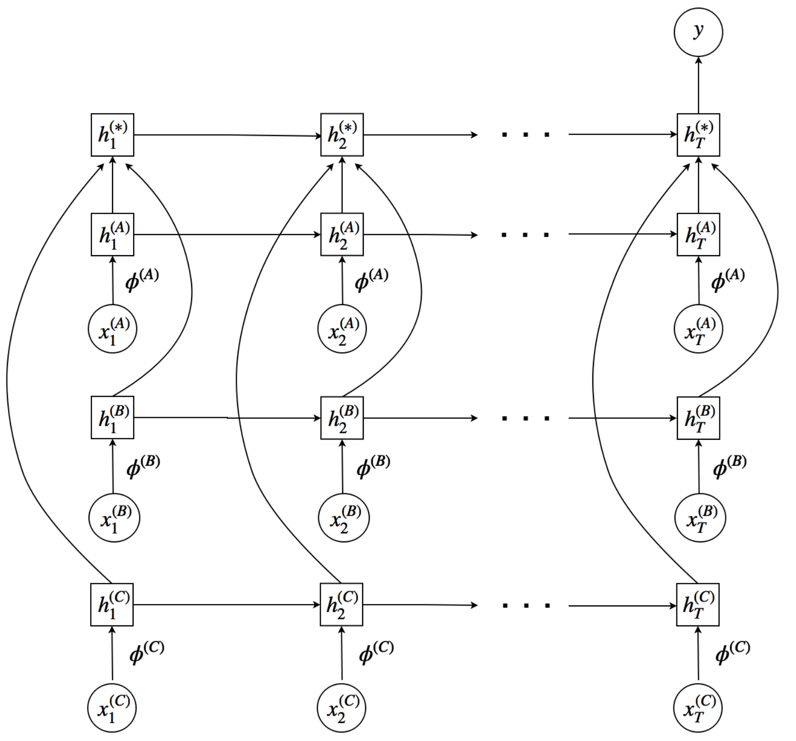
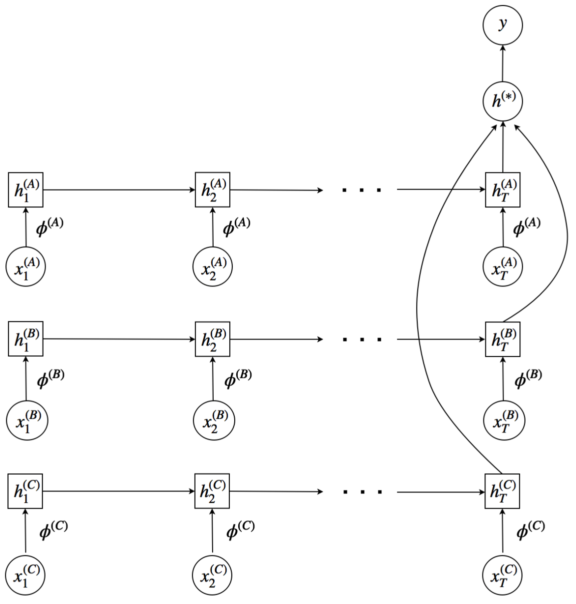
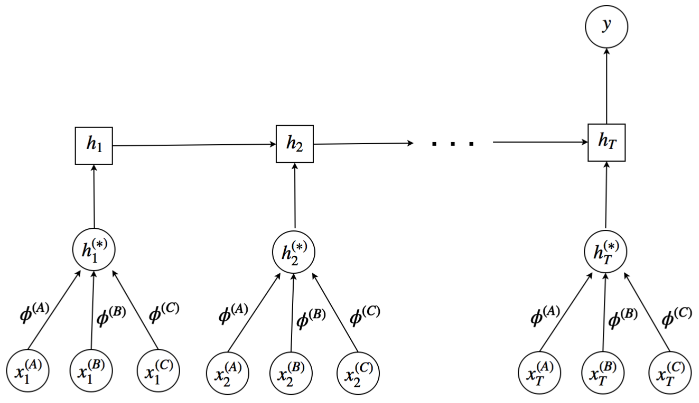


Fig. 2. A graphical representation of the MRNN. The circles represent fully connected layers, the rectangles represent LSTM-RNNs, and the solid lines represent weighted connections. The MRNN has multiple streams of LSTM-RNNs $h_t^{(A)}, h_t^{(B)}, h_t^{(C)}$ that represent the temporal dynamics of input sequences $(x_1^{(A)}, \ldots, x_t^{(A)})$, $(x_1^{(B)}, \ldots, x_t^{(B)})$, $(x_1^{(C)}, \ldots, x_t^{(C)})$ of each modality. We add another LSTM-RNN stream $h_t^{(*)}$ to fuse these modalities while considering temporal dependency $h_{t-1}^{(*)}$. We use appropriate neural network models $\phi^{(A)}, \phi^{(B)}, \phi^{(C)}$ to extract frame-level features from each modality. y denotes a classification result.



(a) Late multimodal fusion model



(b) Early multimodal fusion model

Fig. 3. Alternative multimodal fusion models. **(a)** Late multimodal fusion model fuses modalities $(h_T^{(A)}, h_T^{(B)}, h_T^{(C)})$ to $h^{(*)}$ at only the last step. **(b)** Early multimodal fusion model fuses modalities $(\phi^{(A)}(x_t^{(A)}), \phi^{(B)}(x_t^{(B)}), \phi^{(C)}(x_t^{(C)}))$ to $h_t^{(*)}$ at every step before computing the state h_t of the LSTM-RNN. The important difference between these models **(a)**, **(b)** and the MRNN model (Fig. 2) is that the multimodal fusion processes do not depend on temporal dynamics.

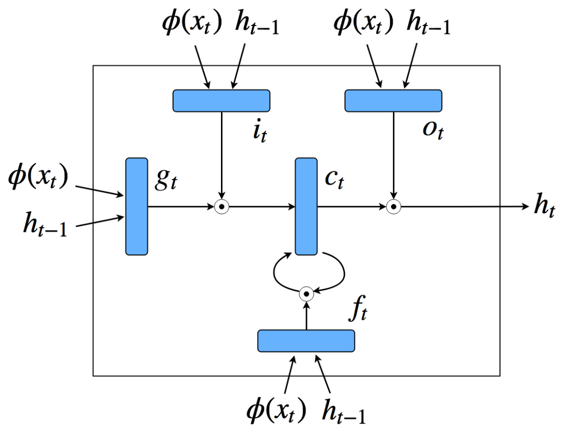


Fig. 4. Graphical representation of an LSTM-RNN. $\phi(\mathbf{x}_t)$ and $\mathbf{h}_{t-1}$ respectively denote a transformed input and the previous state. ϕ is a neural network that extracts the feature vector from the input $\mathbf{x}_t$. $\mathbf{c}_t$, $\mathbf{g}_t$, $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t$ represent a memory cell, an input modulation gate, an input gate, a forget gate, and an output gate, respectively. $\odot$ is element-wise multiplication of vectors.

Here, we omit the bias term for simplicity. We define $\mathbf{h}_0 = \mathbf{0}$. The activation function f (e.g., sigmoidal function and tanh function) is applied to the input vector in an element-wise manner. In this equation, free parameters are the input-to-hidden weight matrix $\mathbf{W}_{in} \in \mathbb{R}^{m \times n}$, the hidden-to-hidden weight matrix $\mathbf{W}_{hh} \in \mathbb{R}^{m \times m}$, and the parameters of ϕ. This equation represents that the hidden state $\mathbf{h}_t$ is dependent not only on the current input $\mathbf{x}_t$ but also on the previous state $\mathbf{h}_{t-1}$. Therefore, $\mathbf{h}_t$ can represent the sequential dynamics of input sequence $(\mathbf{x}_1, \ldots, \mathbf{x}_t)$.

The recurrent structure (Eq. (1)) enables the handling of variable-length sequential data. However, it is known that RNNs tend to suffer from vanishing or exploding gradient problems during training [21]. Because of these problems, RNNs cannot remember long-term dependencies in practice. LSTM-RNNs are an elegant method to solve these problems [12]. LSTM-RNNs have been successfully applied to many applications in natural language processing [22, 23] and speech recognition [24], outperforming conventional sequential models such as HMMs and an Elman RNN. Fig. 4 shows the graphical representation of an LSTM-RNN. An LSTM-RNN is composed of several vectors of same dimension m, a hidden state $\mathbf{h}_t \in \mathbb{R}^m$, a memory cell $\mathbf{c}_t \in \mathbb{R}^m$ and four gates: $\mathbf{g}_t$, $\mathbf{i}_t$, $\mathbf{f}_t$, and $\mathbf{o}_t \in \mathbb{R}^m$. $\mathbf{g}_t$, $\mathbf{i}_t$, $\mathbf{f}_t$ and $\mathbf{o}_t$ denote an input modulation gate, an input gate, a forget gate, and an output gate at step t. At every step, an LSTM-RNN updates its state $\mathbf{h}_t$ using the following equations:

$$\mathbf{g}_t = \tanh(\mathbf{W}_{in}\boldsymbol{\phi}(\mathbf{x}_t) + \mathbf{W}_{hh}\mathbf{h}_{t-1}), \tag{2}$$
$$\mathbf{i}_t = \mathrm{sigmoid}(\mathbf{W}_{ix}\boldsymbol{\phi}(\mathbf{x}_t) + \mathbf{W}_{ih}\mathbf{h}_{t-1}), \tag{3}$$
$$\mathbf{f}_t = \mathrm{sigmoid}(\mathbf{W}_{fx}\boldsymbol{\phi}(\mathbf{x}_t) + \mathbf{W}_{fh}\mathbf{h}_{t-1}), \tag{4}$$
$$\mathbf{o}_t = \mathrm{sigmoid}(\mathbf{W}_{ox}\boldsymbol{\phi}(\mathbf{x}_t) + \mathbf{W}_{oh}\mathbf{h}_{t-1}), \tag{5}$$
$$\mathbf{c}_t = \mathbf{i}_t \odot \mathbf{g}_t + \mathbf{f}_t \odot \mathbf{c}_{t-1}, \tag{6}$$
$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t). \tag{7}$$

In these equations, $\odot$ is an element-wise multiplication of vectors.

Please note that we initialize all the parameters, including those of $\boldsymbol{\phi}$; moreover, we optimize the parameters of $\boldsymbol{\phi}$ simultaneously with the other parameters, using mini-batch stochastic gradient descent (SGD) and backpropagation through time (BPTT) [25, 26].

3.2 Recurrent multimodal fusion

The MRNN is shown in Fig. 2. We develop multiple streams of LSTM-RNNs using ConvNets, the number of which is equal to the number of input modalities. These streams independently update their states according to Eqs. (2)-(7), using input sequences from corresponding modalities. To embed this disconnected information into one common space, we also construct another LSTM-RNN as a fusion stream on top of these streams. At every step, the fusion stream receives the states of the lower streams and fuses them into multimodal-temporal space. Thus, the multimodal fusion process is performed sequentially. Importantly, the fusion structure makes it possible to fuse modalities while considering temporal dependencies. This is the reason we call this fusion strategy "recurrent multimodal fusion". We add a fully connected layer (classification layer) onto the fusion stream. The classification layer receives the last state of the fusion LSTM-RNN. We use the softmax function to compute the probability distribution over categories.

3.3 Late multimodal fusion

As an alternative strategy to incorporate multimodal-temporal data, we propose the late multimodal fusion model shown in Fig. 3 (a). As with the MRNN, the late multimodal fusion model also employs multiple streams of LSTM-RNNs using ConvNets for each input modality. The difference from the MRNN is that the fusion layer is not an LSTM-RNN but a normal fully connected layer. Thus, the multimodal fusion process is performed independently of temporal dependencies. The fusion layer receives the last states of each stream and embeds them into a common space. The last states of each stream hold information about the temporal dynamics of each input modality. Because the fusion process is performed after the LSTM-RNN streams, we call this method "late multimodal fusion". In this sense, this structure is similar to the model of Molchanov et al. (2015) [20], whereas our late multimodal fusion model uses LSTM-RNNs to extract temporal information. As with the MRNN, we add a fully connected layer (classification layer) onto the fusion layer and use the softmax function. The classification layer receives an embedded multimodal feature and predicts the category distribution.

3.4 Early multimodal fusion

The early multimodal fusion model we propose is shown in Fig. 3 (b). This approach integrates multiple modalities using a fully connected layer (fusion layer) at every step before inputting signals into the LSTM-RNN stream. This is the reason we call this strategy "early multimodal fusion". As with the late multimodal fusion model, this model's multimodal fusion process is performed independently of temporal dependencies. This model has only one LSTM-RNN stream, because the input to the LSTM-RNN is already embedded into a common space by the fusion layer. At every step, the LSTM-RNN receives the embedded feature from the fusion layer and updates its state. As with the other two models, the classification layer predicts the probability distribution over categories using the last state of the LSTM-RNN.

4 Experiments

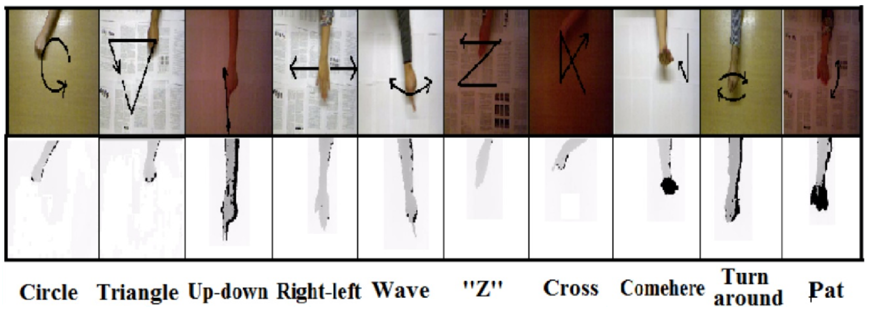


Fig. 5. Examples from the SKIG dataset. This figure is cited from [10].

4.1 Dataset

Using the Sheffiled Kinect Gesture (SKIG) dataset, we compared the MRNN with previous works and our alternative fusion models. The SKIG dataset contains 1080 gesture videos, each belonging to 10 gesture categories. Fig. 5 displays some examples from the dataset. Each video consists of two modalities (a color image sequence and a depth data sequence) captured by the Microsoft Kinect sensor. For preprocessing, we downsized all of the frames to 224×224, and divided them by 255. All videos are collected from six subjects. We devided all videos into three subsets: subject1+subject2, subject3+subject4, and subject5+subject6. We evaluated our models using 3-fold cross validation, in accordance with the previous works.

It is known that optical flow works effectively for action recognition [27]. We believe that adding temporal information (such as optical flow) to the input of the MRNN improves its classification performance. Given an RGB sequence, we computed dense optical flow based on Gunner Farneback's algorithm [28] using OpenCV. To reduce noise, we apply a bilateral filter to every frame before computing optical flow. As a result, we can obtain a two-channel flow data sequence corresponding to the source RGB sequence.

4.2 Network architecture

Throughout our experiments, we fixed the dimension size m of the LSTM-RNNs to 512. We also set the dimension size of the fusion layers to 512. The dimension size of the classification layers is 10, which is equal to the number of gesture categories. We used three modalities in our experiments: color image, optical flow, and depth data. Because these modalities are spatial data, we developed the ConvNet architecture based on Network In Network (NIN) [29] as a nonlinear transformer $\boldsymbol{\phi}$ for each input modality. We show the architecture of the ConvNet that we used in our experiments in Fig. 6. We initialized all parameters (including those of the ConvNets) from a Gaussian distribution, except for hidden-to-hidden weight matrices of the LSTM-RNNs, for which we used an identical matrix according to Le et al. (2015) [30].

4.3 Training settings

We used the cross entropy error as our loss function. We added an L2 regularizatoin term multiplied by 0.0001 to the loss. We used mini-batch SGD with a learning rate of 0.01. After three epochs, we set the learning rate to 0.001. We set the mini-batch size to 5. We computed all gradients of the parameters using BPTT [25, 26]. We implemented the experimental code with Chainer [31], a Python-based open source library for deep learning, on an NVIDIA TITAN X GPU.

4.4 Experimental results

We evaluated the test accuracies of the MRNNs (trained on multiple modalities or a single modality) and our alternative fusion models. We report the results in Table 1. The MRNN trained on multiple modalities significantly outperforms the previous works, and provides improved test accuracy. To the best of our

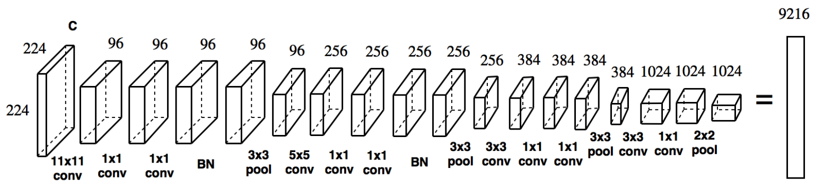


Fig. 6. The architecture of the ConvNet that we used in our experiments. This model has 11 convolution layers (conv), 2 batch normalization layers (BN) [4], and 4 max pooling layers (pool). Each convolution layer is followed by a rectified linear (ReLU) nonlinearity. C is the number of channels of the input frames. We vectorize the last feature maps of size $1024 \times 3 \times 3$ to 9216-dimensional vector.

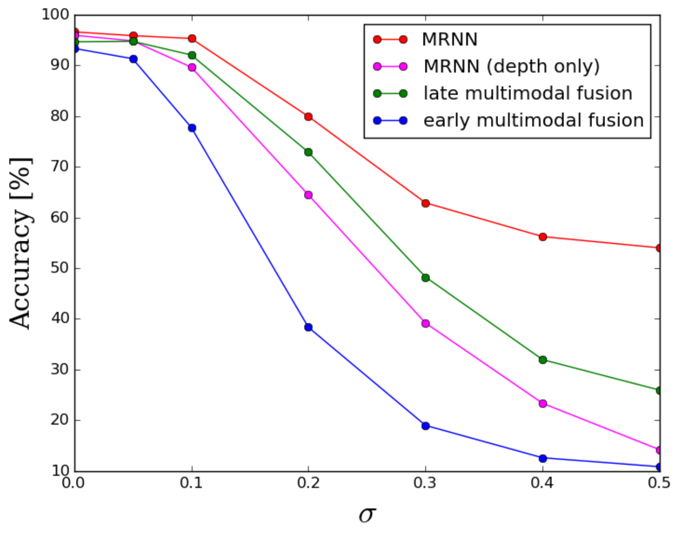


Fig. 7. Test accuracies of the MRNN trained on multiple modalities (red line), the MRNN trained on only depth modality (magenta line), the late multimodal fusion model (green line), and the early multimodal fusion model (blue line). We added Gaussian noise with different standard deviations (denoted by σ) to the depth data of the test inputs. The MRNN trained on multiple modalities successfully utilizes the other modalities to suppress the influence of noise on the depth inputs.

Table 1. Comparison of test accuracy using the SKIG dataset

Method	Accuracy (%)
Liu et al. (2013) [10]	88.7
Choi et al. (2014) [32]	91.9
Tung et al. (2014) [33]	96.7
Early multimodal fusion	94.1
Late multimodal fusion	94.6
MRNN (color only)	91.6
MRNN (optical flow only)	88.5
MRNN (depth only)	95.9
MRNN	**97.8**

knowledge, this accuracy represents the state-of-the-art performance for this dataset. The MRNN trained on multiple modalities also outperforms the other multimodal fusion models. This indicates that a multimodal fusion process that considers temporal dependencies is beneficial for efficient embedding. Compared with the MRNNs trained on a single modality (color only, optical flow only, or depth only), the MRNN trained on multiple modalities also produces better accuracy. Therefore, the MRNN succeeds in utilizing multimodal information effectively.

In our experiments, we also investigated the robustness of the MRNN to noisy inputs. In Fig. 7, we plot the test accuracies when we add Gaussian noise with different standard deviations (denoted by σ) to the depth inputs in the test set. As shown in Fig. 7, the MRNN trained on multiple modalities tends to maintain high accuracy, even when the accuracies of the other models decline. The difference in Fig. 7 indicates that the multimodal fusion structure of ths MRNN provides significant benefits in incorporating modalities to complement each modality.

It is notable that the MRNN does not require any domain-specific localization or segmentation techniques throughout our experiments. The MRNN learns feature extraction, multimodal fusion, and sequential modeling simultaneously in a single architecture in a supervised manner. Therefore, the MRNN is a completely data-driven approach to multimodal gesture recognition, and provides excellent classification performance and high robustness to noisy inputs.

5 Conclusion

In this paper, we proposed a multimodal gesture recognition model that incorporates multiple temporal modalities using multiple streams of LSTM-RNNs. All parameters of the MRNN are optimized towards end-to-end performance in a supervised manner. The MRNN does not require heuristic engineering that is strongly dependent on target gesture domains. We evaluate our recurrent multimodal fusion approach with alternative fusion models. Experimental results indicate that the MRNN (and its multimodal fusion process that considers temporal dependencies) provide significant benefits, and provide excellent classification performance as well as high robustness to noisy inputs. Moreover, the MRNN achieves state-of-the-art performance in the SKIG dataset.

In future, we plan to utilize other modalities such as speech and skeletal data. Moreover, we plan to apply the MRNN to sign language recognition.

6 Acknowledgments

This work was supported by JST CREST, JSPS KAKENHI Grant Number 26730085. We would like to thank the three anonymous reviewers for their valuable feedback on this work.

References

1. Krizhevsky, A., Sutskever, I., and Hinton, G. E.: Imagenet classification with deep convolutional neural networks. In Proc. NIPS (2012)
2. Girshick, R., Donahue, J., Darrell, T., and Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In Proc. CVPR (2014)
3. He, K., Zhang, X., Ren, S., and Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In Proc. ICCV (2015)
4. Ioffe, S., and Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv:1502.03167 (2015)
5. Baraldi, L., Paci, F., Serra, G., Benini, L., and Cucchiara, R.: Gesture recognition in ego-centric videos using dense trajectories and hand segmentation. In Proc. EVW (2014)
6. Darwish, S. M., Madbouly, M. M., and Khorsheed, M. B.: Hand Gesture Recognition for Sign Language: A New Higher Order Fuzzy HMM Approach. Hand, 1:18565 (2016)
7. Bengio, Y., Courville, A., and Vincent, P.: Representation learning: A review and new perspectives. In Trans. PAMI, 35(8):1798-1828 (2013)
8. Wu, J., Cheng, J., Zhao, C., and Lu, H.: Fusing multi-modal features for gesture recognition. In Proc. ICMI (2013)
9. Ohn-Bar, E., and Trivedi, M. M.: Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations. In Trans. ITS, 15(6):2368-2377 (2014)
10. Liu, L. and Shao, L.: Learning discriminative representations from RGB-D video data. In Proc. IJCAI (2013)
11. Elman, J.L.: Finding structure in time. Cognitive science, 14(2):179-211 (1990)
12. Hochreiter, S., and Schmidhuber, J.: Long short-term memory. Neural computation, 9(8):1735-1780 (1997)
13. Murakami, K., and Taguchi, H.: Gesture recognition using recurrent neural networks. In Proc. SIGCHI (1991)

14. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., and Rabinovich, A.: Going deeper with convolutions. In Proc. CVPR (2015)
15. Socher, R., Lin, C. C., Manning, C., and Ng, A. Y.: Parsing natural scenes and natural language with recursive neural networks. In Proc. ICML (2011)
16. Gregor, K., Danihelka, I., Graves, A., and Wierstra, D.: DRAW: A recurrent neural network for image generation. arXiv:1502.04623 (2015)
17. Ji, S., Xu, W., Yang, M., and Yu, K.: 3D convolutional neural networks for human action recognition. In Trans. PAMI, 35(1):221-231 (2013)
18. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In Proc. CVPR (2014)
19. Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T.: Long-term recurrent convolutional networks for visual recognition and description. In Proc. CVPR (2014)
20. Molchanov, P., Gupta, S., Kim, K., and Kautz, J.: Hand Gesture Recognition with 3D Convolutional Neural Networks. In CVPR Workshop on Hand gesture recognition (2015)
21. Bengio, Y., Simard, P., and Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. In Trans. Neural Networks, 5(2):157-166 (1994)
22. Sutskever, I., Vinyals, O., and Le, Q. V.: Sequence to sequence learning with neural networks. In Proc. NIPS (2014)
23. Cho, K., van Merrinboer, B., Bahdanau, D., and Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. In SSST-8 (2014)
24. Graves, A., and Jaitly, N.: Towards end-to-end speech recognition with recurrent neural networks. In Proc. ICML (2014)
25. Werbos, P. J.: Backpropagation through time: what it does and how to do it. In Proc. the IEEE, 78(10):1550-1560 (1990)
26. Rumelhart, D. E., Hinton, G. E., and Williams, R. J.: Learning representations by back-propagating errors. Cognitive Modeling, 5, 3, (1988)
27. Simonyan, K., and Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In Proc. NIPS (2014)
28. Farnebck, G.: Two-frame motion estimation based on polynomial expansion. In Proc. SCIA (2003)
29. Lin, M., Chen, Q. and Yan, S.: Network In Network. In Proc. ICLR (2014)
30. Le, Q.V., Jaitly, N., and Hinton, G.E.: A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. arXiv:1504.00941 (2015)
31. Chainer, `http://chainer.org/`
32. Choi, H., and Park, H.: A hierarchical structure for gesture recognition using RGB-D sensor. In Proc. HAI (2014)
33. Tung, P. T., and Ngoc, L. Q.: Elliptical density shape model for hand gesture recognition. In Proc. ICTD (2014)