LETTER

Low-Cost Method for Recognizing Table Tennis Activity

Se-Min LIM[†], **Jooyoung PARK**[††], ***Nonmembers***, ***and*** **Hyeong-Cheol OH**[†a)], ***Member***

SUMMARY This study designs a low-cost portable device that functions as a coaching assistant system which can support table tennis practice. Although deep learning technology is a promising solution to realizing human activity recognition, we propose using cosine similarity in making inferences. Our experiments show that the cosine similarity based inference can be a good alternative to the deep learning based inference for the assistant system when resources are limited.

key words: *activity recognition, sports skill assessment, wearable technology, cosine similarity, recurrent neural network*

1. Introduction

Wearable technology can be useful for people who practice their table tennis skills. For example, people may not have sufficient knowledge on proper exercises in table tennis but aim to assess their exercise activity in the sport or practice by copying other players' skills, in which case an assistant system with wearable sensors can be helpful [1]. Camera-based approaches are also attractive for this application. This study can provide a preliminary result for the camera-based approaches. Moreover, this study aims to investigate an assistant system which is small enough to be embedded into the wearable sensors for training players on real time. The system receives information from three sensor modules attached to the hand and arm of a table tennis player and recognizes a player's skill that is the closest one, among the ones stored in the system, to the skill which the player hits.

Deep learning (DL) technology is a promising solution to realizing human activity recognition (HAR) [1]–[5]. The use of convolutional neural networks (CNNs) for HAR were studied in [2]–[4]. However, the limits on local connectivity in a CNN prevent the network from effectively dealing with the synchronization problems such as unknown durations or lags between certain points in the time series data. Thus, the long short-term memory (LSTM) recurrent neural networks (RNNs) [6], [7] were proposed for adoption in HAR [5]. In [5], the authors showed that LSTM RNNs functioned better than CNNs as well as conventional machine learning technologies, such as random forest and least-square support vector machine, for HAR problems in everyday life, including walking, running, jumping, sitting, and sleeping. Recently, a coaching assistant method based on the DL technology has been proposed [1].

Cost is a limitation in implementing deep neural networks for portable devices, even for the inference applications only [8]. In the present study, we propose using a low-cost method on the basis of cosine similarity, for recognizing exercise activities in table tennis.

Many prior studies in HAR based on wearable technology use smart phones as data collecting devices. Although a smart phone is very useful for daily monitoring, its size is too large for attachment to a specific point of the human body. Thus, this study uses inertial measurement unit (IMU) sensor modules (MPU-9150s). Figure 1 shows our prototype of the data collection unit. Three IMU modules are wired and send data to a Raspberry Pi 3 model B, which then transmits the data to a notebook via Bluetooth connection. Using the data collected from five players on five table tennis skills, this paper compares the classification performance of cosine similarity with those of unidirectional RNN and unidirectional LSTM RNN. Our experimental results show that unidirectional LSTM RNNs provide the best performance even when the activity time is vague. When the data is preprocessed to further clarify the activity time, the cosine similarity-based method, which can be easily implemented on a small FPGA, shows the classification accuracy of only 2.7% lower than that of the unidirectional LSTM RNN in classifying the five table tennis skills of five players.

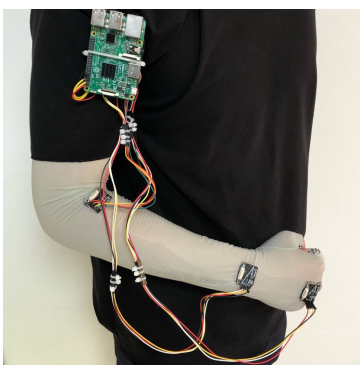


Fig. 1 Data collection unit set up [1].

Manuscript received January 22, 2019.
Manuscript revised May 7, 2019.
Manuscript publicized June 18, 2019.

[†]The authors are with Dept. of Electronic and Information Eng., Korea Univ., SeJong, Korea.
[††]The author is with Dept. of Control and Instrumentation Eng., Korea Univ., SeJong, Korea.
a) E-mail: ohyeong@korea.ac.kr

DOI: 10.1587/transinf.2019EDL8017

2. Background

2.1 Cosine Similarity

For a given feature point k_G in an N-dimensional search space S, the nearest feature point in S can be defined as $argmin_{k \in S} dist(k, k_G)$, where $dist(k, k_G)$ is a distance between a point k in the space and k_G. As a distance measure, the cosine similarity between k and k_G is defined as follows:

$$Cos\_Sim(k, k_G) = \frac{k \cdot k_G}{|k|\,|k_G|} \tag{1}$$

where $\cdot$ denotes an inner product operation.

2.2 Unidirectional LSTM RNN

Figure 2 depicts the operation of a two-level unidirectional LSTM RNN [7], during three time steps, i.e., $t-1$, t, and $t+1$. As described in the level-1 cell at time t, $Cell_t^1$, in Fig. 2, each cell of the LSTM RNN can be modelled with a memory cell and operates at time t as follows [5]:

$$f_t^l = \varphi_f(W_{xf}^l, x_t^l, W_{hf}^l, h_{t-1}^l, b_f^l) \tag{2}$$
$$i_t^l = \varphi_i(W_{xi}^l, x_t^l, W_{hi}^l, h_{t-1}^l, b_i^l) \tag{3}$$
$$o_t^l = \varphi_o(W_{xo}^l, x_t^l, W_{ho}^l, h_{t-1}^l, b_o^l) \tag{4}$$
$$g_t^l = \varphi_g(W_{xg}^l, x_t^l, W_{hg}^l, h_{t-1}^l, b_g^l) \tag{5}$$
$$c_t^l = f_t^l \otimes c_{t-1}^l + g_t^l \otimes i_t^l \tag{6}$$
$$h_t^l = o_t^l \otimes A(c_t^l) \tag{7}$$

where l refers to the network level where the cell is defined, operator $\otimes$ denotes the element-wise multiplication operation, x_t is the input, Ws are the parameter matrices that contain the weights of the network connections, and bs are the biases. The functions φ_f, φ_i, φ_o, and φ_g are called *forget gate*, *input gate*, *output gate*, and *input modulation gate*, respectively; and defined as follows:

$$\varphi_k(W_{xk}, x, W_{hk}, h, b_k) = A(W_{xk}x + W_{hk}h + b_k) \tag{8}$$

where $k = f$, i, o, or g; and $A(\cdot)$ is an activation function. The *internal state* c_t is used to consider the internal recurrence, whereas the *hidden state* h_t is used to consider the outer recurrence. The block labelled with Δ is a memory element to store the internal state.

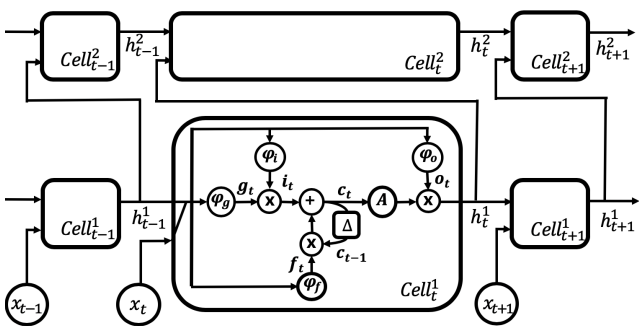


Fig. 2 Two-level unidirectional LSTM RNN [1].

2.3 Unidirectional (Regular) RNN

Using the notation defined in Sect. 2.2, a unidirectional (regular) RNN [6] can be described by the hidden state h_t and the output y_t as follows:

$$h_t^l = A(W_{xh}^l x_t^l + W_{hh}^l h_{t-1}^l + b_h^l) \tag{9}$$
$$y_t = A(W_{hy} h_t^2 + b_y) \tag{10}$$

3. Experiments

3.1 Inference Accuracy

Using the data collection unit shown in Fig. 1, data on five table-tennis skills, namely, forehand long, backhand block, forehand slice, backhand slice, and forehand drive, were obtained from five right-handed players. Each player stroked the ball with each skill 10 times: 7 times for building the models and 3 times for testing them. Each stroke was sampled at 22 time points for 4.4 sec with the frequency of 5 Hz. We only used the tri-axial accelerometers of three sensors in this paper. Thus, 5 persons $\times$ 5 skills $\times$ 3 axes $\times$ 3 sensors $\times$ 7 hits = 1575 data sequences (22 values per sequence) were used to train the neural networks or used as the reference models for making inference based on the cosine similarity.

Two DL models, that is, unidirectional LSTM RNN and unidirectional (regular) RNN, were designed with two levels ($l = 1, 2$) by using the collected data. The number of levels was determined experimentally. The designed LSTM RNN model operated as depicted in Fig. 2. Each level consisted of one LSTM RNN cell. x_t is the input to the first level (cell) at time t, whereas h_t of the first level is the input to the second level. The (regular) RNN model was also designed as described in Eqs. (9) and (10), where x_t was connected to x_t^1, whereas h_t^1 was connected to x_t^2. The rectified linear unit (ReLU) was used as the activation function $A(\cdot)$.

The DL models were trained in the TensorFlow framework [9] by using an Intel i7-7500U CPU and NVIDIA GeForce 930MX GPU. In training the models for classification, a skill was described by a time series of vector (x_t, $t = 0, \cdots, 21$); each vector consisted of up to $3 \times 3 \times 7 = 63$ values. Supervised learning was adopted for classification: the dataset and the label corresponding to the dataset were used together in the training phase. The label was represented with one-hot encoding. The weight and bias were randomly initialized and updated to minimize the cost function. The cost function was the mean cross entropy between the ground truth labels and the predicted output labels. The ground truth labels were the true classes. An Adam optimizer was used as the optimization algorithm to minimize the cost function. Training was performed with data obtained from the accelerometer sensors after the minmax-scaling by the MinMaxScaler of sklearn [10]. The batch size

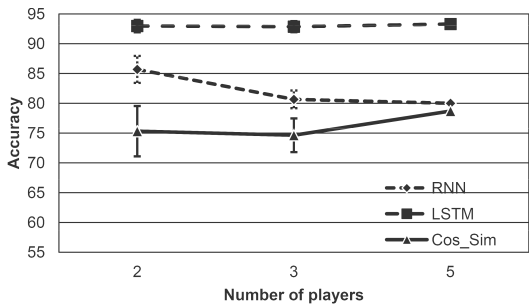
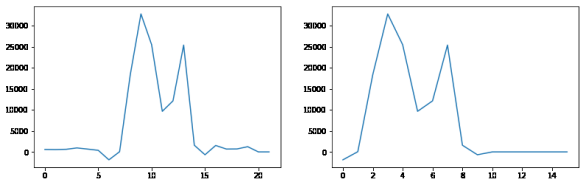
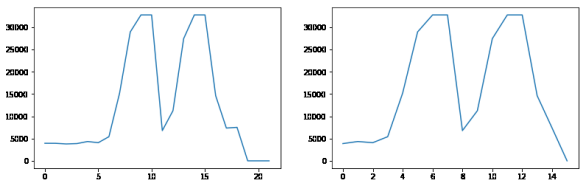


Fig. 3 Classification accuracy for original data.



(a) *skill-1*(Original) (b) *skill-1*(Preprocessed)



(c) *skill-4*(Original) (d) *skill-4*(Preprocessed)

Fig. 4 Data preprocessing. Meaningful data points are extracted from the minimum in the beginning interval to the minimum in the ending interval. (a)(b) When the number of meaningful data points is less than 16, zeros are added. (c)(d) When the number of meaningful data points exceeds 16, the first 15 data points followed by one zero are used.

and the number of hidden units were empirically found after some tuning for improved accuracy. L2 regularization was used to prevent network overfitting, whereas the dropout technique was not adopted.

For making inference on the basis of cosine similarity, one action (hit) was described by a vector (k in Eq. (1)), with a length of $3\times3\times22$. Three actions for each of the 5 persons $\times$ 5 skills were compared with $5 \times 5 \times 7 = 175$ stored actions, from which the maximum cosine similarity was used for making inference.

Figure 3 presents the classification accuracy achieved by the three considered methods for the data originally collected from the sensors. For two or three players, 10 combinations out of five players were tested for 10 or 15 labels for each combination, respectively. In the figure, the average values are connected by lines, whereas the standard deviations are depicted by error bars. The results show that the LSTM RNN outperforms the cosine similarity by 14.6%, for the five-player case.

In the data originally collected from the sensors, the time and duration of the activities vary from player to player and from skill to skill. For example, the action does not constantly begin at time point 6 in Fig. 4 (a). The LSTM RNN works robustly even when the activity time is vague, but the

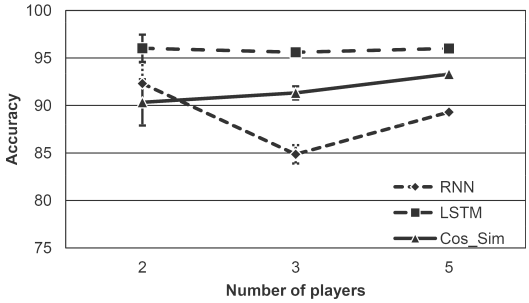


Fig. 5 Classification accuracy for preprocessed data.

Table 1 Number of operations required for classification.

	LSTM	RNN	Cos. Sim.
Multiplication	11,975,400	4,774,000	75,000
Addition	12,079,200	4,550,000	70,875
Square Root	0	0	3,150
Division	0	0	1,575

regular RNN and the cosine similarity do not. However, fortunately for this application, the meaningful part of the data points that are collected during the activity time only can be easily extracted by detecting the changes in the accelerometer value. Thus, we added a simple preprocessing step which is depicted in Fig. 4: The length of a data sequence is reduced to 16 data points. The meaningful data points are extracted from the minimum in the beginning interval to the minimum in the ending interval. When the number of meaningful data points is less than 16 as in (a), zeros are added as in (b). When the number of meaningful data exceeds 16 as in (c), the first 15 data points followed by one zero are used as in (d). Figure 5 presents the inference accuracy achieved after the preprocessing step. Using the preprocessed data, the accuracy of all three methods increased, although only a small change was observed for the LSTM RNN; as it was noted in [5], the model captured a relatively long-range dependencies in the input sequence. The accuracy of the cosine similarity-based method improved significantly; its accuracy was notably better than that of the regular RNN and only 2.7% less than that of the LSTM RNN, for the five-player case. The decrease in accuracy with the regular RNN on the three-player case is due to the players' characteristics – due to the similarity in the skills of the newly added player to those of other players.

3.2 An FPGA Implementation

Table 1 summarizes the number of operations required for inference by the three considered methods. The number of operations required by the LSTM RNN is significantly larger than that of the cosine similarity-based one. Considering that the processing time of the LSTM RNN strongly depends on its hardware implementation, we only evaluate the performance of the cosine similarity based method in this study.

Given that the data preprocessing step can be imple-

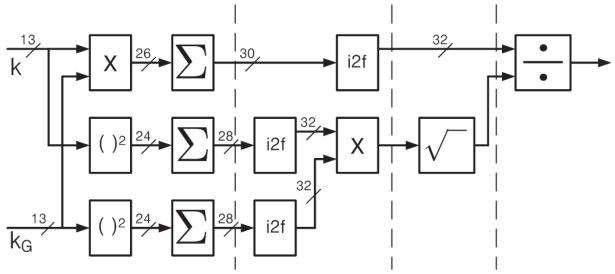


Fig. 6 Block diagram of the cosine similarity calculator. The broken lines represent the higher-level pipelining registers.

Table 2 Implementation results. The maximum operation frequencies shown are estimated with the Slow 1100mV 85°C model.

Logic utilization [ALMs]	Register [FFs]	DSP [Blocks]	Block memory [bits]	Max. Op. freq. [MHz]
945	2,306	7	4,751	176

mented using a simple hardware for searching local minima, the cosine similarity calculator is the most expensive part of the proposed system. This section presents our design of the cosine similarity calculator and evaluates its FPGA implementation. Figure 6 depicts the block diagram of our design. In the figure, "∑"s and "i2f"s denote the accumulators and integer-to-floating-point converters, correspondingly. The accuracy results of the proposed system, presented in Sect. 3.1, were obtained when sensor data were represented as 13-bit signed integers. Thus, a 13-bit signed integer multiplier and two 12-bit unsigned squarers [11] were used in the input stage. For simplicity, the modules for changing the operands to their absolute form, at the input ports of the squarers, are omitted in the figure.

The designed calculator was modelled in Verilog and implemented with Quartus Prime 18.0 with targeting an Intel Cyclone FPGA (5CSEMA5F31C6 [12]) using Intel megafunctions [13] and the squarer in [11]. Table 2 summarizes the implementation results and shows that only a small portion of the FPGA was consumed.

The designed calculator operates with a two-level pipelining scheme. The broken lines in Fig. 6 represents higher-level pipelining registers which operates at a 22-division frequency of the lower-level clock. Although each activity data sequence consists of 16 values after preprocessing, the squarers and the multiplier are pipelined. Thus, we use 22 lower-level clocks to process one higher-level pipeline stage. With this structure, the proposed design can compare a skill activity with the stored 1575 skill activities in approximately 0.2 ms, when the data are provided in time.

4. Conclusions

We proposed a low-cost method on the basis of cosine similarity for recognizing exercise activities in table tennis and compared its performance with those of two well-known DL solutions in HAR. Our experiments have shown that a unidirectional LSTM RNN provides the best inference accuracy as expected. However, the classification accuracy is only 2.7% lower in the proposed method than in the unidirectional LSTM RNN, in classifying the five table tennis skills of five players. The proposed method can be implemented in a small FPGA area and work sufficiently fast.

Only a relatively small number of data points from accelerometers were used in this study to reduce implementation cost. Future research may tackle topics about using other sensors and collecting more data points to improve inference accuracy. Issues related to the hardware implementation of assistant systems based on LSTM RNNs can also be investigated in future studies.

Acknowledgments

This work was supported in part by a Korea University Grant. The authors would like to thank the members of CompMath group, Korea Univ. for their help with collecting data.

References

[1] S.-M. Lim, H.-C. Oh, J. Kim, J. Lee, and J. Park, "LSTM-guided coaching assistant for table tennis practice," Sensors, vol.18, 4112, 2018.
[2] M. Zeng, L.T. Nguyen, B. Yu, O.J. Mengshoel, J. Zhu, P. Wu, and J. Zhang, "Convolutional neural networks for human activity recognition using mobile sensors," Proc. Int. Conf. on Mobile Computing, Applications, and Services, pp.197–205, 2014.
[3] Y. Chen and Y. Xue, "A deep learning approach to human activity recognition based on single accelerometer," Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, pp.1488–1492, 2015.
[4] H.-O. Hessen and A.J. Tessem, "Human activity recognition with two body-worn accelerometer sensors," Master's Thesis, Norwegian Univ. of Sci. & Tech., Trondheim, Norway, 2015.
[5] A. Murad and J.-Y. Pyun, "Deep recurrent neural networks for human activity recognition," Sensors, vol.17, pp.2556–2071, 2017.
[6] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, and J. Schmidhuber, "A novel connectionist system for improved unconstrained handwriting recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol.31, no.5, pp.203–215, 2008.
[7] H. Zen and H. Sak, "Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis," Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing, pp.4470–4474, 2015.
[8] K. Guo, S. Zeng, J. Yu, Y. Wang, and H. Yang, "A survey of FPGA based neural network inference accelerators," ACM Trans. Reconfig. Tech. & Syst., vol.12, no.1, Article 2, April 2018.
[9] M. Abadi et al., "TensorFlow: A system for large-scale machine learning," Proc. USENIX Conf. on Operating Systems Design and Implementation, pp.265–283, 2015.
[10] "scikit-learn:Machine Learning in Python," http://scikit-learn.org/stable/
[11] S. Choi and H.-C. Oh, "Pipelined squarer for unsigned integers of up to 12 bits," IEICE Trans. Inf. & Syst., vol.E101-D, no.3, pp.795–798, March 2018.
[12] Intel Corp., "Cyclone V Device Overview," CV-51001, https://www.intel.com, accessed 2018.
[13] Intel Corp., "IP and Megafunctions," https://www.intel.com/content/www/us/en/programmable/support/literature/lit-ip.html