Machine Learning and Applications: An International Journal (MLAIJ) Vol.6, No.2/3, September 2019

Human Activity Recognition Using Recurrent Neural Network

Yoshihiro Ando

yoshihiro.ando@gmail.com

Abstract. With the spread of smartphones incorporating various sensors, accelerometers and gyro sensors have become familiar to us. Based on such situations, sensor-based human activity recognition (HAR) that uses human sensor data to identify human activity has come to use smartphones as data acquisition sources. In the early studies of HAR using smartphones, handcrafted methods were used if various statistical values were required as feature quantities and high accuracy was realized. Meanwhile, the popularization of deep learning in recent years has not been discussed, and its application has been made to HAR. Although deep learning has the advantage of being able to automatically extract feature quantities from data, it has not reached a step beyond precision in handcrafted methods. Furthermore, in the previous research, to divide data by time window of a fixed interval, except for some part, inference could not be performed unless the data for the time window was secured. We attempted to overcome these limitations using recurrent neural network. Our method records higher accuracy than previous studies using convolutional neural network and long short term memory, which are typical methods in deep learning and display results comparable to handcrafted methods. We also succeeded in pre-calculating many feature quantities, whose calculation was a problem in the previous research, and eliminating the time window.

1 Introduction

Video-based and sensor-based human activity recognition (HAR) techniques identify human behavior using image and sensor data. With the spread of smartphones with various built-in acceleration and gyro sensors, smartphones have come to be used for sensor-based HAR.

Research on HAR using sensor data from smartphones has a history of nearly 10 years. Anguita, D. et al. [2] calculated 561 feature quantities from the accelerometer and gyro sensor data of a smartphone and realized high accuracy by using the input value to the support vector machine (SVM).

In recent years, different methods of deep learning are used in HAR. For example, methods using convolutional neural network (CNN) by Ronao, C. A. and Cho, S. B. [3] and Jiang, W. and Yin, Z. [4] and methods using recurrent neural network (RNN) by Inoue, M. et al. [5].

Methods using deep learning have recorded high accuracy such as the method using long short-term memory (LSTM) by Zhao, Y. et al. [6]. However, such a method using deep learning has not reached a step further in terms of accuracy compared to the so-called handcrafted method by Anguita, D. et al.

Deep learning differs from such handcrafted feature amount computation because it has the advantage of being able to automatically extract the feature amount. Therefore, to take advantage of such features, an improvement in the performance of HAR accuracy by deep learning is desirable.

Furthermore, in the previous research, data is divided by time window into model time series data except for some parts. When dividing by time window, a disadvantage that arises is that inference cannot be performed until data satisfying the time window is obtained. For example, this is a problem in cases where the delay causes a fatal inconvenience for safety such as an estimation of the driver's behavior while driving a car.

We propose a method to perform normalization processing only on the data of the acceleration sensor and the gyro sensor. We can then use the result as an input vector to the RNN. This is to omit many characteristic quantity calculations and to avoid dividing the data by time window. Furthermore, our method recorded higher accuracy than previous studies using CNN and LSTM.

2 Related research

2.1 Multi-class Support Vector Machine

Anguita, D. et al. [2] identified six types of motion from smartphone sensor data using multi-class support vector machine (MC-SVM).

They attached the smartphones on the subject's waist and acquired the values of acceleration sensors and gyro sensors for six types of motion such as walking and ascending and descending stairs. Next, the acquired data was divided by a time window of 2.56 seconds. Values, such as average value, maximum value, and entropy, for each of the XYZ axes were calculated and used as the SVM feature vector. The data separated by the time window was created so that 50% of the previous and the next overlapped. By calculating 561 features, they obtained a high accuracy of 96%. The SVMs they used corresponded to multi-class classification by OVA (One vs. All).

Table 1 shows the attributes of the data set used by them, and Table 2 shows the breakdown of the six operations.

Nine types of values are used for sensor data as shown in Table 1. They are as follows: acceleration without considering gravity (Linear Acceleration) for axial values of XYZ; acceleration considering gravity (Total Acceleration) for axial values of XYZ; and angular velocity value of rotational axis (Angular Velocity) for axial values of XYZ.

Table 1. Data set from Anguita, D. et al. [2]

Number of people	30
Age	19 to 48
Device used	SAMSUNG Galaxy S II
Sampling rate	50Hz
Motion recording time per person	7 to 8.7 minutes
Type of sensor	acceleration, angular velocity
Type of motion	6
Number of training data	21 people (data number 470,528)
Number of test data	9 people (data number 188,608)

Table 2. Six types of action

Walking
Walking upstairs
Walking downstairs
Sitting
Standing
Lying

Anguita, D. et al. [2] have published data sets used for research, and we used their data set in the evaluation of the proposed method.

2.2 Deep Convolutional Neural Network

Jiang, W. and Yin, Z. [4] proposed a method for processing sensor data with deep convolutional neural network (DCNN).

Their method first converts sensor data into an image called "Signal Image (SI)." Next, the SI is processed by the Discrete Fourier Transform (DFT) into an image called "Activity Image (AI)." Finally, AI is processed by DCNN.

The DCNN of Jiang, W. and Yin, Z. [4] consists of an input layer that inputs AI, an hidden layer, and an output layer that outputs inference results. The hidden layer, in turn, contains five layers: two convolutional layers, two subsampling layers, and one fully connected layer.

By its nature, this method generates and uses an image of fixed size. Therefore, processing cannot be started without data corresponding to the length of the time window. Jiang, W. and Yin, Z. [4] set the length of the time window to 68. In addition to this, the method has the disadvantage that processing such as imaging and DFT must be performed before processing by DCNN.

This method presents 95.18% as the evaluation result for the data set of Anguita, D. et al. [2]. This result does not exceed 96% of Anguita, D. et al. [2].

2.3 Residual Bidirectional Long Short-term Memory

Zhao, Y. et al. [6] proposed a method called residual bidirectional long short-term memory (Res-Bidir-LSTM) in which network paths called redidual and bidirectional were added to LSTM.

LSTM mitigates the vanishing or exploding gradient problem that occurs when learning long series data. The vanishing gradient problem refers to the phenomenon in which the gradient gradually disappears or scatters when back propagation of a multi-layered network or back propagation of long time series data in the reverse time direction occurs.

Zhao, Y. et al. [6] added two methods to this LSTM. The first is the residual method that skips the processing of some hidden layers and adds the input value for that layer to the input value of the next layer as it is. The second is the bidirectional method that performs time series data processing not only in the forward direction but also in the reverse direction. Both methods contribute to improving the accuracy of deep learning using neural networks.

The structure of the network is inevitably complicated because the networks of Zhao, Y. et al. [6] have two special "paths" inherent in the network as described above. In addition, they divide the 50 Hz data by the time window which length is 128 steps and process it like the method of Anguita, D. et al. [2]. This also has the disadvantage that inference cannot be performed unless data is accumulated for a certain period.

Zhao, Y. et al. [6] present 93.58% as a result of processing the data set of Anguita, D. et al. [2] with Res-Bidir-LSTM. Their results do not exceed 96% of Anguita, D. et al. [2].

3 Proposed method

Our proposed method aims to solve the two problems that exist in the previous studies described so far. One of the two problems is that the real-time nature of inference is impaired by the time window. The other problem is that SVM using handcrafted features is more accurate than deep learning.

This chapter describes the details of the proposed method.

3.1 Data pre-processing

In our proposed method, processing other than normalization is not performed as data pre-processing. Each of the training and test data is processed as one continuous data for each person. Because there are nine types of sensor data in each of these continuous data, nine elements of the input vector are arranged in time series.

Labels for the six types of motion assigned to the data are assigned to each input vector. In our proposed method, training data and test data are not separated by motions and processed continuously. This is because it is possible to identify a smartphone that acquires sensor data. However, when processing real-world data, human actions are not prelabeled. Therefore, we assume that it is necessary to process acquired data continuously.

In addition, in our proposed method, both numerical calculation, such as maximum value and entropy, and data imaging are not performed. Furthermore, the data is not divided according to the time window.

As per the above-mentioned training and test data, special pre-processing is not required and implementation becomes easy. In addition, by not performing division by the time window, it is possible to perform inference continuously without waiting for a certain amount of data to be collected.

3.2 Normalization

Perform normalization as data pre-processing as per Equation 1.

$$x'_{ni} = \frac{x_{ni} - \overline{x_i}}{\sigma_i} \tag{1}$$

Here, x'_{ni} represents the n-th data of the i-th input unit and x_{ni} represents the n-th data corresponding to the i-th input unit. $\overline{x_i}$ represents the mean value of the data corresponding to the i-th input unit. σ_i represents the standard deviation of the data corresponding to the i-th input unit. $\overline{x_i}$ and σ_i are calculated only for training data. For test data, we use the values calculated for training data. This is because when inferring the actual data in real time after implementing the proposed method, you cannot know in advance the mean value or standard deviation of that data.

3.3 Initial value of weight

In our proposed method, we use probability variables (normal random numbers) according to the standard normal distribution generated by the Box-Muller method shown in Equation 2 for the network weight $\mathbf{w}$.

$$r = \sqrt{-2 \log r_1} \cos 2\pi r_2 \tag{2}$$

Here, r_1 and r_2 are independent of each other, and both represent random numbers uniformly distributed on $(0, 1)$, and r represents the generated random number. In Equation 2, r follows a normal distribution with mean 0.0 and standard deviation 1.0. Equation 3 is used to change the mean and standard deviation.

$$r' = \mu + \sigma r \tag{3}$$

Here, μ is the mean, σ is the standard deviation, r is the random number generated by Equation 2, and r' is the random number we use.

In RNN, it is impossible to directly determine the global minimum of the error function. In the proposed method, the random numbers are used to generate the uniformly distributed random numbers r_1 and r_2 in Equation 2 to obtain a good local minimum. The seed of the random number is varied within a certain range, and the seed with the best accuracy is adopted.

3.4 Division of data

In the previous research, sensor data was divided by time window but not in the proposed method. Figure 1 shows the data division in the previous research. By contrast, Figure 2 shows the data division in the proposed method.

It is common in HAR to divide the data into certain intervals and use it as an input vector (e.g., in the method of Anguita, D. et al. [2]). By contrast, in the proposed method, data is not divided in a fixed interval. This makes it possible to process data sequentially. In addition, because the number of elements of the input vector only needs to correspond to

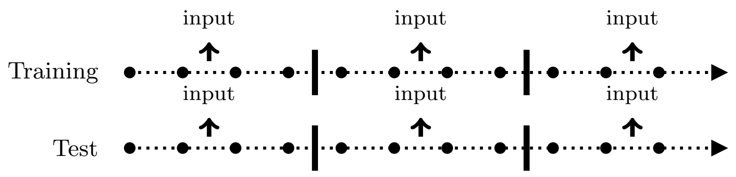

Fig. 1. Time window in conventional method.

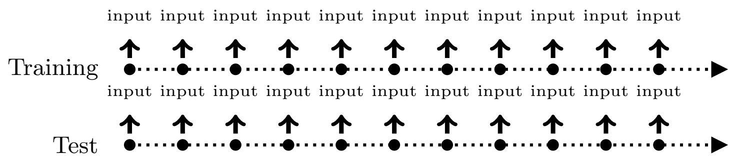

Fig. 2. Time window in proposed method.

the type of sensor data, the number of units constituting the network is reduced. For example, while the number of units in the input layer is $68 \times 36 = 2448$ in DCNN of Jiang, W. and Yin, Z. [4], only nine corresponding to the values of each sensor are sufficient in the proposed method.

3.5 Network structure

The structure of RNN in the proposed method is illustrated in Figure 3. The unit in the hidden layer is provided with a loop line to add the output of the unit to the next input. Since there are many units in the hidden layer, some of the units are omitted in Figure 3.

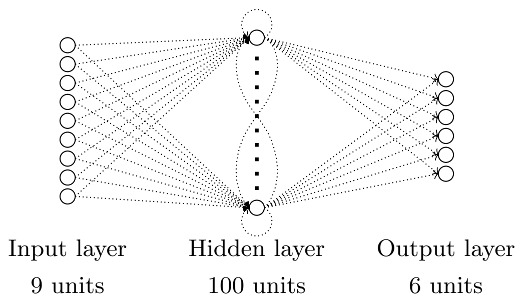

Fig. 3. Structure of RNN in the proposed method.

The nine units of the input layer correspond to three types of Linear Acceleration X, Y, and Z; three types of Total Acceleration X, Y, and Z; and three types of Angular Velocity (Gyro) X, Y, and Z. This makes it in total nine types of Signal Sequences. The number of units in the hidden layer is determined after some trial and error. The six units in the output layer correspond to six types of actions.

3.6 Activation function

The "identity map" shown by Equation 4 is used for the activation function of the input layer in the proposed method.

$$y_i = x_i \tag{4}$$

Here, x_i represents the input value of the i-th unit and y_i represents the output value of the i-th unit.

Use the "Sigmoid" function shown in Equation 5 for the activation function of the hidden layer.

$$y_i = \frac{1}{1+\exp(-x_i)} \tag{5}$$

Here, x_i represents the input value to the i-th unit and y_i represents the output value from the i-th unit.

Use the "softmax" function shown in Equation 6 for the activation function of the output layer.

$$y_i = \frac{\exp(x_i)}{\sum_{j=1}^{J}\exp(x_j)} \tag{6}$$

Here, x_i is the input value of the i-th unit, y_i is the output value of the i-th unit, and J is the number of units in the output layer.

3.7 Error function

The error function in the proposed method uses the cross entropy of Equation 7.

$$E(\mathbf{w}) = -\sum_{n}\sum_{i} d_{ni}\log y_{ni}(\mathbf{x}_n;\mathbf{w}) \tag{7}$$

Here, $\mathbf{w}$ is the weight of the network, d_{ni} is the value that the i-th output unit should output for the n-th data, and the value of d_{ni} is 0 or 1. y_{ni} represents the output value of the i-th output unit for the n-th data, and $\mathbf{x}_n$ represents the n-th data.

The gradient descent method shown in Equation 8 is used to obtain the minimum value of Equation 7.

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \epsilon \nabla E_n \tag{8}$$

Here, $\mathbf{w}^{t+1}$ is the weight of the network at time $t+1$, $\mathbf{w}^t$ is the weight of the network at time t, ϵ is the learning coefficient, and ∇E_n is the gradient of the error function for the n-th data.

Back propagation through time (BPTT) is used to update the weight of the loop connection.

3.8 Learning coefficient

Fixed learning coefficients are hardly used in recent years; thus, our proposed method uses AdaGrad [7], shown in Equation 9, as learning coefficients.

$$-\frac{\epsilon}{\sqrt{\sum_{t'=1}^{T} g_{t',k}^2}} g_{t,k} \tag{9}$$

Here, ϵ is the pre-defined fixed learning coefficient, T is the last time of the series data, $g_{t',k}$ is the gradient used to update the k-th weight $w_{t',k}$ at time t', and $g_{t,k}$ is the gradient used to update the weight $w_{t,k}$ at time t.

4 Evaluation

4.1 Hyperparameter

Table 3 shows hyperparameters in the proposed method.

Table 3. Hyperparameters used for training

Number of units in hidden layer	100
Learning coefficient	0.1 (AdaGrad)
Mini batch	0.3 times random extraction
Initial weight	Normal random number
Mean	0
Standard deviation	0.01
Maximum epoch	2000
Sampling rate	12.5Hz

The optimal model for the test data is searched by changing the random number sequence (random number seed). In our evaluation, 20 kinds of random number sequences are used. The original sampling rate of the data set is 50 Hz. In the proposed method, we reduced it to 1/4 to increase the training speed.

4.2 Error curve and accuracy

Figure 4 is a plot of training and test errors and accuracies.

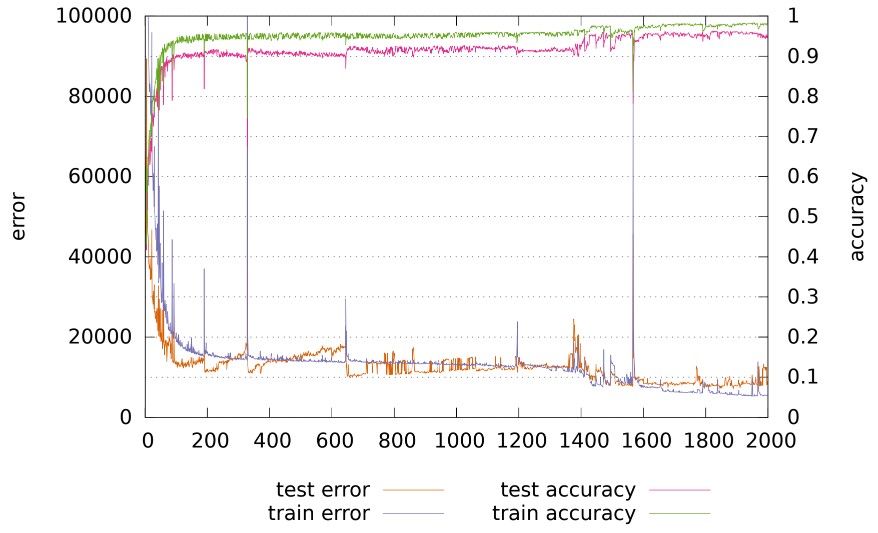

Fig. 4. Error and accuracy in each epoch of training

The graph in Figure 4 is somewhat zigzag because we use a mini batch.

4.3 Results

Table 4 compares the accuracy of the previous research with the proposed method.

5 Conclusions

We have shown that the proposed method can achieve high accuracy without using handcrafted features in HAR. At the same time, we have

Table 4. Evaluation results of prior research and proposed method

Method	Accuracy
MC-SVM(Angita, D.)	96.37%
DCNN(Jiang, W.)	95.18%
Res-Bidir-LSTM(Zhao, Y.)	93.58%
RNN(Proposed)	**96.25%**

shown that the proposed method can be inferred without being tied to the time window. We have shown that we can get higher accuracy than deep learning method in previous research.

References

1. P. Casale, O. Pujol, and P. Radeva, "Human activity recognition from accelerometer data using a wearable device," in *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 2011, pp. 289–296.
2. D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, "A public domain dataset for human activity recognition using smartphones." in *Esann*, 2013.
3. C. A. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," *Expert systems with applications*, vol. 59, pp. 235–244, 2016.
4. W. Jiang and Z. Yin, "Human activity recognition using wearable sensors by deep convolutional neural networks," in *Proceedings of the 23rd ACM international conference on Multimedia*. Acm, 2015, pp. 1307–1310.
5. M. Inoue, S. Inoue, and T. Nishida, "Deep recurrent neural network for mobile human activity recognition with high throughput," *Artificial Life and Robotics*, vol. 23, no. 2, pp. 173–185, 2018.
6. Y. Zhao, R. Yang, G. Chevalier, X. Xu, and Z. Zhang, "Deep residual bidir-lstm for human activity recognition using wearable sensors," *Mathematical Problems in Engineering*, vol. 2018, 2018.
7. J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.