applied sciences

MDPI

Article

Fall Detection System Based on Simple Threshold Method and Long Short-Term Memory: Comparison with Hidden Markov Model and Extraction of Optimal Parameters

Seung Su Jeong [1], Nam Ho Kim [2] and Yun Seop Yu [1,*]

[1] ICT & Robotics Engineering, Semiconductor Convergence Engineering, AISPC Laboratory, IITC, Hankyong National University, 327 Jungang-ro, Anseong-si 17579, Gyenggi-do, Korea
[2] Department of Embedded System, Bundang Convergence Technology Campus of Korea Polytechnic, 5, Hwangsaeul-ro 329 beon-gil, Sengnam 13590, Gyeonggi-do, Korea
* Correspondence: ysyu@hknu.ac.kr

Abstract: In an aging global society, a few complex problems have been occurring due to falls among the increasing elderly population. Therefore, falls are detected using a pendant-type sensor that can be worn comfortably for fall detection. The sensed data are processed by the embedded environment and classified by a long-term memory (LSTM). A fall detection system that combines a simple threshold method (STM) and LSTM, the STM-LSTM-based fall detection system, is introduced. In terms of training data accuracy, the proposed STM-LSTM-based fall detection system is compared with the previously reported STM-hidden Markov model (HMM)-based fall detection system. The training accuracy of the STM-LSTM fall detection system is 100%, while the highest training accuracy by the STM-HMM-based one is 99.5%, which is 0.5% less than the best of the STM-LSTM-based system. In addition, in the optimized LSTM fall detection system, this may be overfitted because all data are trained without separating any validation data. In order to resolve the possible overfitting issue, training and validation data are evaluated separately in 4:1, and then in terms of validation data accuracy of the STM-LSTM-based fall detection system, optimal values of the parameters in LSTM and normalization method are found as follows: best accuracy of 98.21% at no-normalization, no-sampling, 128hidden layer nodes, and regularization rate of 0.015. It is also observed that as the number of hidden layer nodes or sampling interval increases, the regularization rate at the highest value of accuracy increases. This means that overfitting can be suppressed by increasing the regularization, and thus an appropriate number of hidden layer nodes and a regularization rate must be selected to improve the fall detection efficiency.

Keywords: fall detection; the elderly; long short-term memory (LSTM); overfitting; regularization

Citation: Jeong, S.S.; Kim, N.H.; Yu, Y.S. Fall Detection System Based on Simple Threshold Method and Long Short-Term Memory: Comparison with Hidden Markov Model and Extraction of Optimal Parameters. *Appl. Sci.* **2022**, *12*, 11031. https://doi.org/10.3390/app122111031

Academic Editor: Hanatsu Nagano

Received: 30 August 2022
Accepted: 26 October 2022
Published: 31 October 2022

1. Introduction

It was reported by the World Health Organization (WHO) that 30% of the population aged 65 or older suffer from falls more than once a year and this percentage increases to 50% in the population aged 80 or older [1,2]. The fall-related consequences are serious injuries (e.g., femoral neck fracture, brain damage, skin burns), which in most cases lead to physical and cognitive disability or death in cases of undetected falls [2–4]. To reduce these risks, such as death rate and severity of injuries, by detecting falls as quickly as possible, researchers are now focusing on developing automatic fall detection systems that generate alerts when events occur [5–7].

Studies on fall detection of the elderly with wearable devices [8–10] or smartphones [11–15] using 3-axial accelerometer have been reported. The devices with 3-axial accelerometer are attached to the body to distinguish between falls and activities of daily living (ADL). Among the existing studies, a simple threshold method (STM) and machine learning methods for fall detection of the elderly were reported [16–23]. The STM is weak against

noise, and it is difficult to distinguish it from patterns relatively similar to falls such as lying or sitting. Machine learning methods, such as hidden Markov model (HMM) [17,18], layered hidden Markov model (LHMM) [19], decision tree (DT) [20], K-nearest neighbor (KNN) [21,22], and support vector machine (SVM) [22,23], provided a higher accuracy in fall detection than the STM. In the case of HMM, there are many problems in obtaining a temporally aligned sequence and ensuring that the data satisfy a fixed distribution [24]. SVM has the disadvantage of increasing resource consumption and slowing down as the number of classifications increases [25]. To compensate for these shortcomings, research using deep neural networks was needed. Accordingly, there is an increasing number of reports on research on fall detection systems that apply deep neural networks to wearable devices [26–31]. Gated recurrent unit (GRU) [27], recurrent neural network (RNN), long short-term memory (LSTM) [26–31], and LSTM combined with convolutional neural network (LSTM-CNN) [30,31] have been studied using 3-axis accelerometer data for fall detection in the elderly. In most studies using LSTM, and GRU, attempts have been made to adjust the LSTM network, combine various networks, and change the type of sensors [32–34]. In LSTM-CNN, methods, such as 1-dimensional (1D) or 2-dimensional (2D) convolution depth and layer addition, were used [34–37]. RNNs suffer from the gradient vanishing problem [38], and LSTM and GRU are variants of traditional RNN, which is proposed for solving the problem of RNN. As GRU does not contain many trainable parameters, accuracy and computation speed of GRU are relatively lower and faster than those of LSTM, respectively [27]. As the network of LSTM-CNN is relatively complicated, the computation speed can be slower than that of LSTM and GRU [32–35]. Therefore, because LSTM is a simple model to predict time-series fall data in terms of accuracy and computation speed, LSTM-based fall detection systems have been reported [26–31]. Although rapid detection is required due to the nature of fall, several LSTM fall detection models [26–31] have relatively slow fall detection. Moreover, several LSTM-based fall detection systems have not performed any normalization [27–31]. In addition, there are concerns about actual operation without any validation process [26–29,31], and thus there is a possibility that the training data will overfit. Most LSTM-based fall detection systems applied raw data to LSTM, not feature parameters, and further research concerning, e.g., z-score normalization following a Gaussian distribution, and regularizations to prevent overfitting and sampling methods of rescaling data for the best fall detection has not been investigated [26–31].

Power efficiency is very important for wearable devices, which are embedded devices with limited resources. To increase the power efficiency of wearable devices to which a fall detection system using deep neural networks is applied, a fall detection algorithm with good power efficiency is required. A study has been published to increase power efficiency by combining a fall detection method using a simple threshold and HMM, one of machine learning (STM-HMM) [17]. HMM predicts the next event with respect to the previous event, therefore it may be unsuitable for multiple continuous data such as fall data [24]. In this STM-HMM, even changing to deep neural networks instead of HMM may improve both fall detection accuracy and power efficiency. Accordingly, it is necessary to study a fall detection system that applies LSTM, deep learning only the data exceeding the threshold of the 3-axis acceleration sensor data or its pre-processing data. Because the training data and the verification data were not tested separately in the fall detection using the existing HMM [17], it was not confirmed whether the training data were overfitted. In a fall detection system using STM-HMM, it is necessary to divide training data into training and validation data to determine overfitting and prevent it. When applying LSTM to a fall system, it is also necessary to investigate the optimal conditions to increase the accuracy.

In this study, a fall detection system combining the STM with LSTM (STM-LSTM) is proposed. The 3-axis accelerometer data are calculated with several parameters, and then the STM is processed. For this fall detection system, the LSTM is applied only when the parameter exceeds the threshold to detect the fall events. Section 2 describes the materials used in this study. Section 3 explains the proposed STM-LSTM for fall detection. In Section 4, the performance of fall detection for the SMM-LSTM is compared against that

for the STM-HMM, and in Section 5, optimal parameters in the proposed STM-LSTM are investigated. Finally, we present the conclusion of this study.

2. Materials

2.1. Edge Device

A fall-detection system which can classify falls and ADLs of a person by applying a parameterized dataset to STM-HMM has recently been reported [17]. The dataset was acquired from a self-developed embedded edge device that was attached as a pendant to the participant's neck and was used as sensor node [17]. The device consists of a ±8 g 3-axial accelerometer (BMA150, Bosch) and Zigbee wireless communication module (CC2430, Texas Instrument). A gateway was used to collect data from multiple wireless sensor nodes. A server was used to classify falls and ADLs by applying the parameters calculated from the 3-axial acceleration data to the proposed fall-detection algorithm. Figure 1 shows a photo of participants wearing the real device for implementing the proposed fall detection system, respectively. A detailed description of this device is described in Ref. [17].

Figure 1. Photo of a person wearing a real edge device as a pendant to participant's neck for implementing the proposed fall detection system.

2.2. Dataset

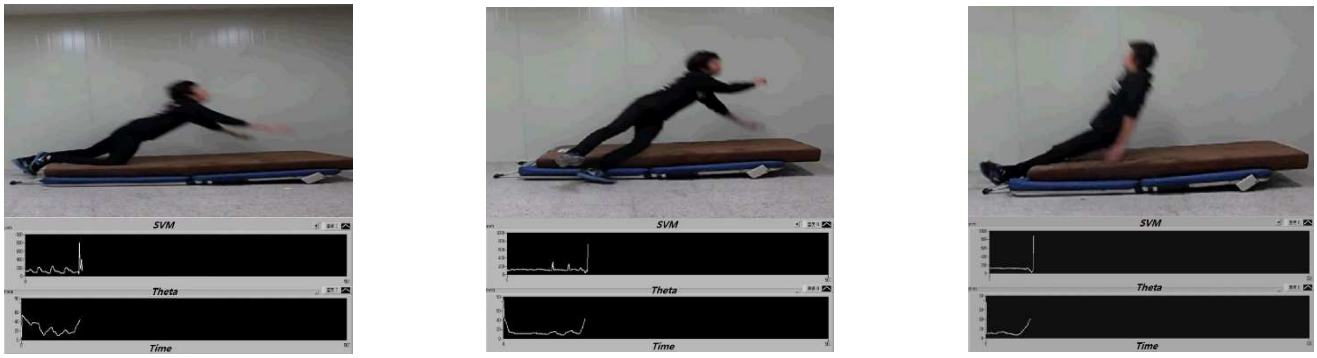
The experiment was performed by six healthy subjects, consisting of four men and two women aged 20–50, 160–185 cm tall, and 50–85 kg in weight. To distinguish between falls and activities of daily living (ADL), the subjects performed four types of ADLs and three types of falls, as shown in Table 1.

Table 1. Description of four types of ADL and three types of fall.

Activities		Description
ADLs	ADL-a	Walking
	ADL-b	Lying
	ADL-c	Running
	ADL-d	Jumping
Falls	Fall-a	Falling forward
	Fall-b	Falling sideway
	Fall-c	Falling backward

The three types of falls were performed using a mattress with a thickness of 20 cm for the safety of the subjects. It was tested using a total of 560 data consisting of 320 ADLs and

240 falls. The number of activities of subjects A, B, C, and D (age: 20) was 15, and those of E (age: 50) and F (age: 40) were 10. Figure 2 shows the photos performing 3 types of fall and their parameters, sum vector magnitude (SVM), and angle (θ) calculated from measured 3-axial acceleration data. It shows a sudden increase in SVM and angle at the moment of falling.



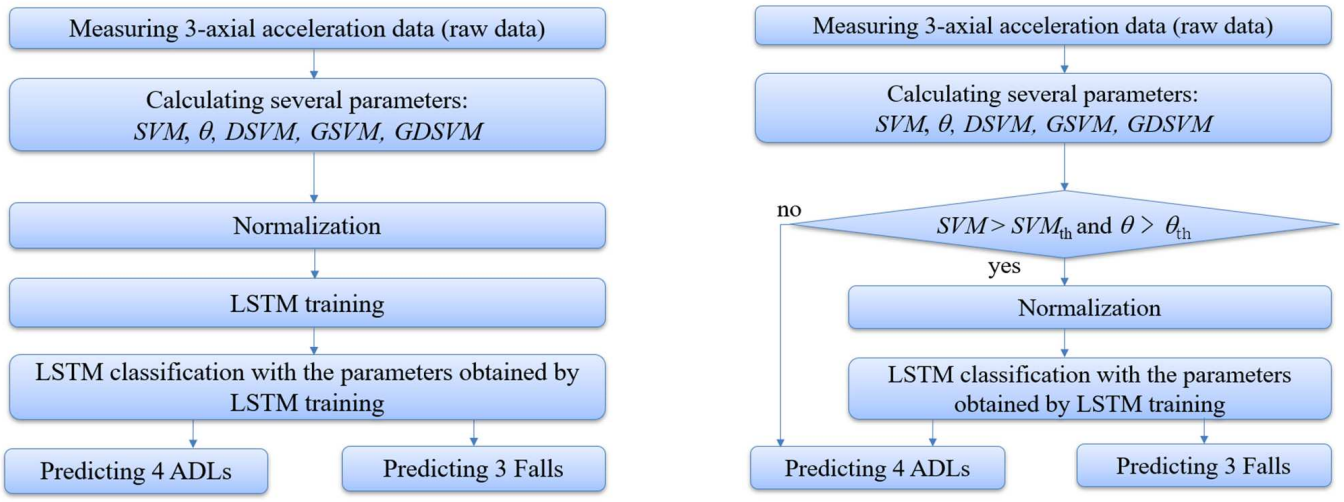
(**a**) (**b**) (**c**)

Figure 2. Photos performing 3 types of fall and their measured parameters, sum vector magnitude (*SVM*) and angle (Theta; θ). (**a**) Falling forward, (**b**) Falling sideway, (**c**) Falling backward.

3. Method

3.1. Algorithm of Fall Detection System

Figure 3a,b show the flow charts of training and test modes of the proposed fall detection system, respectively. In both training and test modes, the data acquired from the edge device are converted into various parameters (see Section 3.2). In the training mode, the converted parameters are normalized, and then all parameters are learned using LSTM. Using the results learned by the LSTM, 3 types of falls and 4 types of ADLs are distinguished. In the test mode, the first possible fall is primarily determined through STM. The STM means that it is determined as a fall event when *SVM* and θ exceed the thresholds of *SVM* (SVM_{th}) of 2.5g and θ (θ_{th}) of 46.42°, respectively [17]. Then, the first determined fall data are just classified according to 3 types of falls and 4 types of ADLs using LSTM learned in training mode.



(**a**) (**b**)

Figure 3. Flow charts of the proposed fall detection system. (**a**) Training mode and (**b**) test mode.

The reason for using the STM for the test mode is to more efficiently control power problems in edge device environments with limited resources. Because power consumption

of the edge device, including the Zigbee wireless communication module, is dominant at the transmission event, the transmission must be reduced. Because the edge devices are not suitable for applying deep neural network such as LSTM due to its limited computation resource, the converted parameters must be transmitted to the server. If the STM is used, the first determined fall data can just be transmitted to the server instead of transmitting all data. In general, fall-like situations that exceed the thresholds of SVM and θ have a very low frequency, and thus better computational and power efficiency is expected.

3.2. Parameters

The data measured from the edge device are converted into 5 types of parameters, which are θ, SVM, differential SVM ($DSVM$), and gravity-weighted SVM ($GDSVM$), and gravity-weighted $DSVM$ ($GDSVM$). They are calculated as follows [17]:

$$\theta(i) = \tan^{-1}\left(\frac{\sqrt{m_y^2(i) + m_z^2(i)}}{m_x(i)}\right) \times \frac{180}{\pi}, \quad (1)$$

$$SVM(i) = \sqrt{m_x^2(i) + m_y^2(i) + m_z^2(i)}, \quad (2)$$

$$DSVM(i) = \sqrt{(m_x(i) - m_x(i-1))^2 + (m_y(i) - m_y(i-1))^2 + (m_z(i) - m_z(i-1))^2}, \quad (3)$$

$$GSVM(i) = \frac{\theta(i)}{90} \times SVM(i), \quad (4)$$

$$GDSVM(i) = \frac{\theta(i)}{90} \times DSVM(i), \quad (5)$$

where i represents the sampling number, and $m_x(i)$, $m_y(i)$, and $m_z(i)$ represent the x-axial, y-axial, and z-axial acceleration of the ith sampling, respectively. For STM, SVM and θ are used. The parameters are finally used as the average value using the sliding window method [39] in which the modified nth parameter is calculated as the average from n-69 to n parameters as follow as one example (θ):

$$\bar{\theta}(n) = \frac{1}{70}\sum_{i=n-69}^{n} \theta(i). \quad (6)$$

For the input data of LSTM in the fall detection system, single and multiple parameters are used. The single parameters, P_θ, P_S, P_D, P_G, and P_{GD} represent the average values of 70 θs, 70 SVMs, 70 $DSVM$s, 70 $GSVM$s, and 70 $GDSVM$s, respectively, as expressed in Equation (6). The multiple parameters, $P_{\theta S}$, $P_{\theta D}$, $P_{\theta G}$, $P_{\theta GD}$, and P_{ALL}, represent the combination of P_θ and P_S, P_θ and P_D, P_θ and P_G, P_θ, and P_{GD}, and P_θ, P_S, P_D, P_G, and P_{GD}, respectively, as shown in Table 2.

Table 2. Combination of single parameters for double and multiple paramters.

Parameters		Combination
Double parameters	$P_{\theta S}$	$P_\theta + P_S$
	$P_{\theta D}$	$P_\theta + P_D$
	$P_{\theta G}$	$P_\theta + P_G$
	$P_{\theta GD}$	$P_\theta + P_{GD}$
Multiple parameters	P_{ALL}	$P_\theta + P_S + P_D + P_G + P_{GD}$

3.3. Normalizations

Using the data acquired from the 3-axial accelerometer, the parameterized data are calculated, and normalization process such as the Min-Max and Z-score normalizations [16]

are performed to minimize the parameterized data. In the min-max normalization, the range of the parameterized data is rescaled as the range in [0, 1], which is given by

$$X_{Min-Max} = \frac{x - x_{min}}{x_{max} - x_{min}}, \tag{7}$$

where x is the original parameterized data, $X_{Min\text{-}Max}$ is the Min-Max normalized data, and x_{min} and x_{max} are the smallest and largest numbers in each data set, respectively. The Z-score normalization makes the parameterized data have zero-mean and unit-variance, which is given by

$$X_{Z-score} = \frac{x - x_{mean}}{x_{std}}, \tag{8}$$

where $X_{Z\text{-}score}$ is the *Z-score* normalized data, and x_{mean} and x_{std} are the mean and standard deviation in each data set, respectively. Each normalization is processed before training on the LSTM.

3.4. Proposed LSTM Network

3.4.1. Overview of LSTM

To overcome the difficulties in training the RNN model due to gradient vanishing [38] and error blowing up, the LSTM, in which nonlinear units are replaced in conventional RNNs, were proposed. Figure 4 shows the typical structure of an LSTM cell. An LSTM cell contains a self-connected memory cell and three gates, namely the forget-gate, the input-gate, and the output-gate [40]. The three gate units are the essential components to learn the long-term patterns by preventing memory contents from irrelevant inputs and outputs. The input-gate controls the flow of input activations into the cell-memory, forget-gate controls how much information will flow from the cell-memory, and the output gate controls the output flow of cell activations into the rest of the network. The cell computes the cell output h_t and the updated cell-memory output c_t from the previous cell outputs (h_{t-1}), the previous cell-memory outputs (c_{t-1}), and the sequential input x_t at time step t as follows:

$$f_t = \sigma_g(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f), \tag{9}$$

$$i_t = \sigma_g(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i), \tag{10}$$

$$o_t = \sigma_g(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o), \tag{11}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \sigma_h(W_{xc}x_t + W_{hc}h_{t-1} + b_c), \tag{12}$$

$$h_t = o_t \odot \sigma_h(c_t), \tag{13}$$

where W_{xf}, W_{hf}, W_{xi}, W_{hi}, W_{xo}, and W_{h0} are the weight matrices from the input to the forget-gate, from the previous output to the forget-gate, from the input to the input-gate, from the previous output to the input-gate, from the input to the output-gate, and from the previous output to output-gate, respectively, W_{cf}, W_{ci}, W_{co} are the diagonal weight matrices for peephole connections, and b_f, b_i and b_o are the bias vectors of the forget-gate, the input-gate, and the output-gate, respectively. $\odot$ denotes the Hadamard product, and σ_x and σ_g denote the hyperbolic tangent and sigmoid functions, respectively. i_t, f_t, and o_t are the input-gate, forget-gate, and output-gate, respectively.

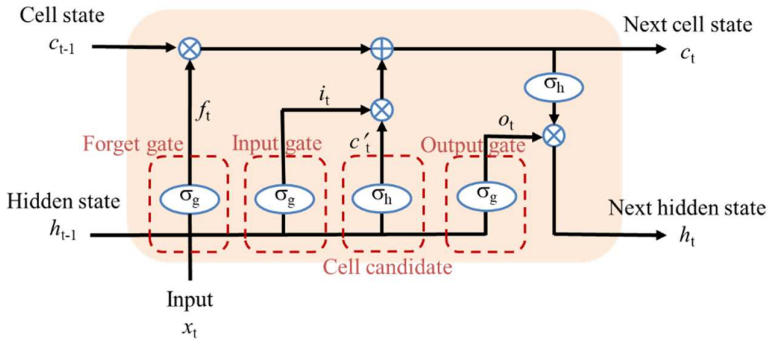


Figure 4. The typical structure of long-short term memory (LSTM). σ_h and σ_g denote the hyperbolic tangent and sigmoid functions, respectively.

3.4.2. LSTM Network

Figure 5 shows the proposed LSTM network for the fall detection. It consists of one input layer, two hidden layers, and dense layer. The input size of the input layer represents batch × length × dimension of data. In the input layer, batch means the number of all input data including falls and ADLs. Because 3-axial accelerometer acquired accelerations as 100 Hz during 5 sec, the length of data consists of 500 by default, and the length can be shortened to $500/sn$ when sampling with sampling interval sn. In addition, the data dimension is 1, 2, and 5 for single, double, and multiple parameters, respectively. One hidden layer consists of n LSTMs, and the number of hidden nodes (n) in each hidden layer can be changed for performance optimization. The dense layer includes the softmax and optimization, and the output is displayed as 7 result values in the dense layer. The result value is represented using the one hot encoding method.

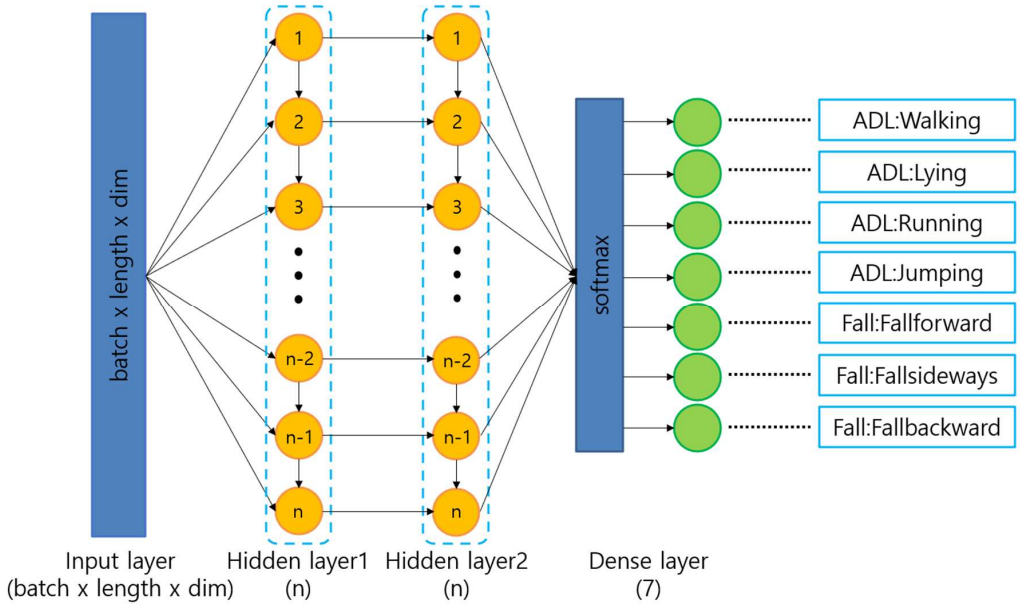


Figure 5. Fall detection model architecture for LSTM.

3.4.3. Regularization in Loss Function

There are several types of loss functions: mean squared error (MSE), root mean squared error (RMSE), binary cross entropy (BCE), and categorical cross entropy (CCE). Among them, the CCE with softmax used as a common loss function for multi-class classification [41] is selected. One requirement for using CCE is that the labels of the output should follow the one-hot encoding method. The CCE is expressed as [42]

$$L_{CCE} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} p_{ij}\log\left(q_{ij}\right) \quad (14)$$

where N is the size of the data set, C is number of classes, p_{ij} is the true probability distribution which the ith training pattern belongs to jth category, and q_{ij} is the predicted probability distribution for ith observation belonging to class j. L2 regularization [43] is added to the CCE loss function to address overfitting issues. The regularized cost function, $J(W)$, is expressed as [44,45]

$$J(W) = L_{CCE} + \frac{\lambda}{2}\|W\|_2, \tag{15}$$

where W is the connection weight between all layers, λ is the regularization rate, and $\|W\|_2$ represents the L2 norm. Parameter λ determines a trade-off between the training error and the generalization ability [44].

3.5. Classification

Accuracy (ACC), sensitivity (SEN), and specificity (SPE) are used as common evaluation indicators for classification [46] as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \times 100\%\,, \tag{16}$$

$$SEN = \frac{TP}{TP + FN} \times 100\%, \tag{17}$$

$$SPE = \frac{TN}{TN + FP} \times 100\%, \tag{18}$$

where true positive (TP), false negative (FN), true negative (TN), and false positive (FP) are the numbers of correctly classified falls, ADL classified falls, correctly classified ADLs, and fall classified ADLs, respectively. Accuracy, sensitivity, and specificity refer to the probabilities of accurately predicting all ADL and fall, fall of all falls, and ADL of all ADLs, respectively.

3.6. Experiment Environment

The platform for training LSTM is the Anaconda for Jupyter Notebook, and the software environment is Keras 2.8.0 and Tensorflow 2.8.0 on Windows 10 and hardware NVIDIA RTX3080 GPU is used for training. As a second platform for training LSTM, a Linux environment was also used for learning, a Jupyter Notebook on Centos Linux 7 was used for software, and NVIDIA Quadro RTX6000 GPU was used for hardware.

4. Comparison of HMM and LSTM

4.1. Setting of LSTM

An STM-HMM-based fall detection system has been reported to perform well at ACC of 99.5% [17]. The proposed STM-LSTM-based fall detection system shown in Figure 3 are almost same to the STM-HMM-based fall detection system except for using LSTM for training and testing all data instead of HMM. All 560 data are trained with both HMM and LSTM, and then all of them are tested without separating any validation data. Therefore, they may be overfitted. In this section, the validity of the proposed STM-LSTM-based fall detection system is investigated through a comparison with the STM-HMM-based fall detection system, although both systems are overfitted. In Section 5, the proposed STM-LSTM-based fall detection system is evaluated by training data and test data separately at 4:1, and different normalizations, regularization rates, numbers of hidden node, and sampling numbers in the LSTM are compared to obtain the optimal conditions for the excellent fall detection.

In test mode of the proposed STM-LSTM-based fall detection system, it is firstly determined as a fall event when SVM and θ exceed SVM_{th} of 2.5g and θ_{th} of 46.42°, respectively, as shown in Figure 3b. Then, the first determined fall data are just classified with 3-types of falls and 4-types of ADLs using LSTM training model learned in training mode. Table 3 shows the parameters for the LSTM training model of the proposed STM-

LSTM-based fall detection system. No normalization is applied, and in the input layer node, the batch, length, and dimension of data are 560, 500, and 1 (for single parameters), 2 (for double parameters), and 5 (for quintuple parameter), respectively. The number of hidden layer and output layer nodes is 13 and 7, respectively, and the learning rate and regularization rate is 0.0025 and 0.00015, respectively.

Table 3. LSTM training model parameters and methods.

Parameters		Values
Learning rate		0.0025
Input layer nodes	Batch	560
	Length	500
	Dimension	1, 2, 5
Number of hidden layer nodes		13
Regularization rate		0.00015
Number of output layer nodes		7

4.2. Comparison of STM-HMM and STM-LSTM-Based Fall Detection System

Figure 6 shows the confusion matrix applying five types of single parameter to the proposed STM-LSTM-based fall detection system. The reason why there is no walking pattern in the confusion matrix except for the single parameter P_S is that the LSTM inference is not applied if the threshold value is not reached, as shown in the algorithm in Figure 3b. Applying the single parameter P_G achieves the best accuracy, sensitivity, and specificity of 100%. In Table 4 and Figure 7, applying the single parameters, the training data performance of the proposed STM-LSTM-based fall detection system is compared with that of the previously reported STM-HMM-based fall detection systems. In Figure 7, the solid and dashed lines denote training data performances of the STM-LSTM and STM-HMM-based fall detection systems, respectively, and the squares, circles, and triangles denote accuracy, sensitivity, and specificity, respectively. This shows the training data performance calculated using the network parameters obtained from the training data (i.e., all 560 data). The best performance of the STM-HMM-based fall detection system is when applying the single parameter P_θ, while that of the STM-LSTM-based fall detection system is when applying the single parameter P_G. For all single parameters except for the case of the single parameter P_θ, the performance of the STM-LSTM-based fall detection system is relatively better than that of the STM-HMM-based fall detection system, and the best performance of fall detection is achieved in the STM-LSTM-based fall detection system.

Figure 8 shows the training data performance of the proposed STM-LSTM-based fall detection system, applying the multiple parameters. Applying the multiple parameter P_{ALL} achieves the best accuracy, sensitivity, and specificity of 100%. The squares, circles, and triangles denote accuracy, sensitivity, and specificity, respectively. The average performance with the multiple parameters shows relatively higher accuracy than that with the single parameters. All 560 data are trained with both HMM and LSTM with the algorithm in Figure 3a, and all of them are tested without separating any validation data with the algorithm in Figure 3b. Therefore, they may be overfitted to the extent that accuracy of 100% is obtained for the single parameter P_G and the multiple parameter P_{ALL}. In order to resolve the possible overfitting issue, training and validation data should be evaluated separately in 4:1.

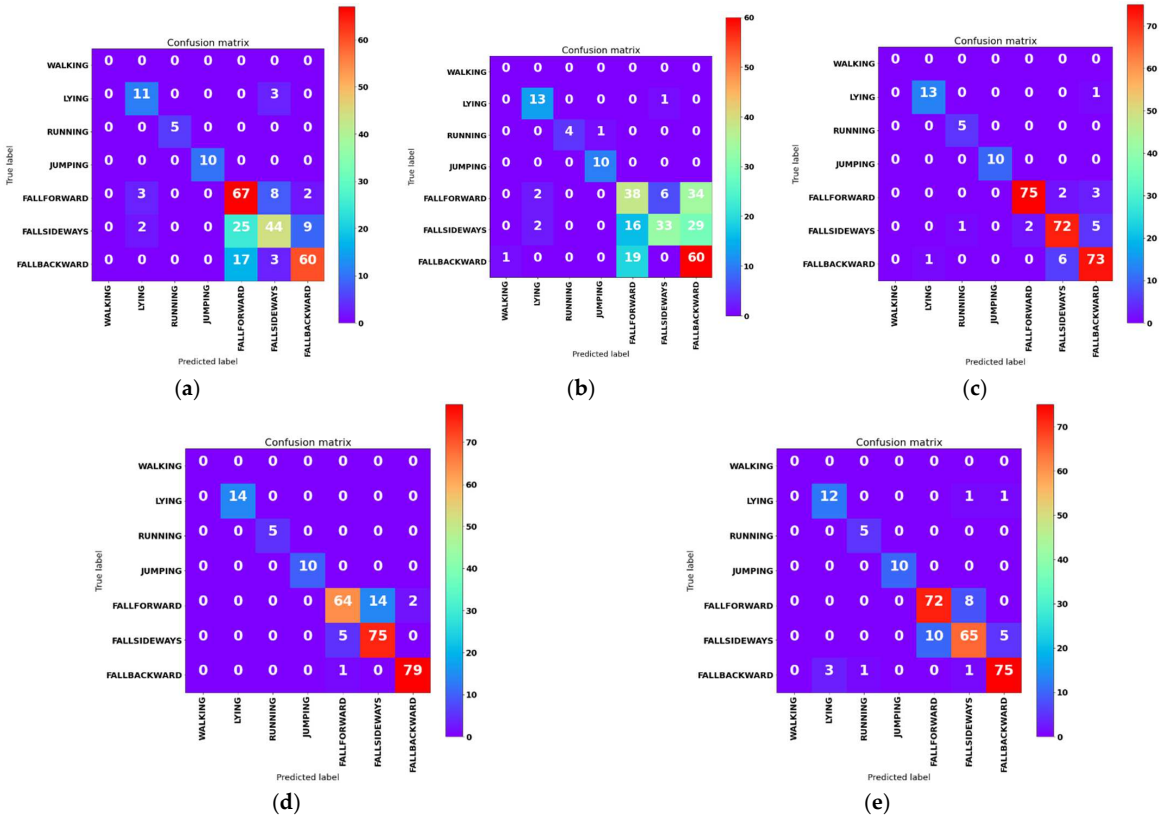


Figure 6. Confusion matrix applying 5-types of single parameter to the proposed LSTM-based fall detection system. (**a**) P_θ, (**b**) P_S, (**c**) P_D, (**d**) P_G, (**e**) P_{GD}.

Table 4. Fall detection results of the proposed STM-LSTM-based and the previously reported STM-HMM-based fall detection systems, applying the single parameters.

	Accuracy [%]		Sensitivity [%]		Specificity [%]	
	LSTM	**HMM**	**LSTM**	**HMM**	**LSTM**	**HMM**
P_θ	98.57	99.5	99.06	99.17	97.91	99.96
P_S	98.21	96.43	99.31	97.5	96.66	95.63
P_D	99.46	98.57	99.68	99.6	99.16	97.81
P_G	100	97.86	100	99.17	100	96.88
P_{GD}	98.92	98.21	99.37	99.17	98.33	97.5

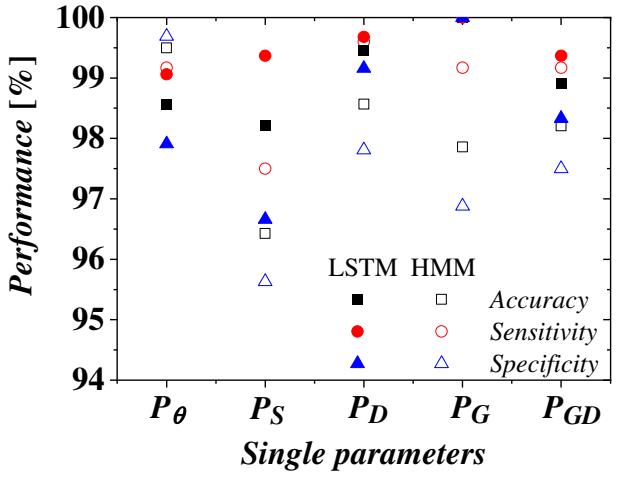


Figure 7. Training data performances of the STM-LSTM and STM-HMM-based fall detection systems, applying the single parameters.

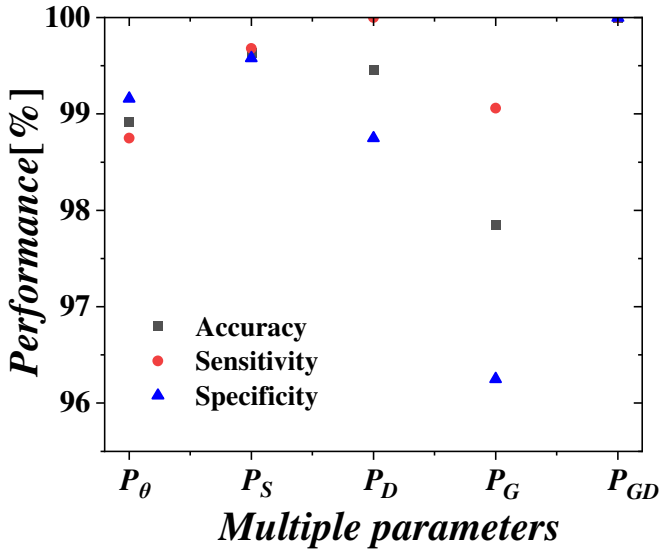


Figure 8. Training data performance of the STM-LSTM-based fall detection system, applying the multiple parameters.

5. Optimization of LSTM for Fall Detection System

The previously published STM-HMM-based fall detection system [17] was not verified against untrained data. It should be verified that the data obtained from the experiment can be used for actual fall detection. Therefore, it is necessary to use a part of the training data as data for verification. Training data and validation data follow a ratio of 4 to 1, and validation data are not included in training. The validation data performance calculated from the validation data using the network parameters obtained from the training data is investigated. Parameter values are as shown in Table 4, and the single parameter P_θ, no additional sampling, and no normalization are set by default. Since P_θ showed the best performance as a single parameter in the STM-HMM-based fall detection system [17], the optimal values of parameters used in LSTM and normalization method are investigated for P_θ. The investigating methods and parameter ranges are shown in Table 5.

Table 5. Model configuration.

Model Configuration	Parameter Ranges/Methods
Normalization	No-normalization, Min-Max, Z-score
Regularization rate, λ	0.00015 to 0.65
Sampling interval, *sn*	1, 3, 5, 7, 9
Multiple data dimension	Single parameter P_θ
Number of hidden layer nodes, *n*	6, 13, 32, 64, 128, 256

5.1. Normalization

In this section, the impact of different normalization process of the calculated parameters, including No-normalization, Min-Max normalization, and Z-score normalization, is examined. Figure 9 shows the validation data performance of the STM-LSTM-based fall detection system when applying No-normalization, Min-Max normalization, and Z-score normalization. Accuracies applying No-normalization, Min-Max normalization, and Z-score normalization are 95.93%, 92.85%, and 86.6%, respectively. The best performance is shown when no normalization is applied, and the best accuracy, sensitivity, and specificity are 95.93%, 93.75%, and 97.91%, respectively. It showed the worst accuracy when the Min-Max normalization is applied. To investigate the reason why accuracies of the Min-Max or Z-score normalizations are lower than that of No-normalization, time series data of a running pattern and a forward fall pattern before and after the Min-Max or Z-score normalizations are investigated, as shown in Figure 10. In Figure 10, black and red

lines denote running and forward fall patterns, respectively. After the Min-Max or Z-score normalizations, the running and forward fall patterns are so similar to each other that it is difficult to easily distinguish them, and thus the accuracies by the Min-Max or Z-score normalizations can be reduced, compared to that of No-normalization.

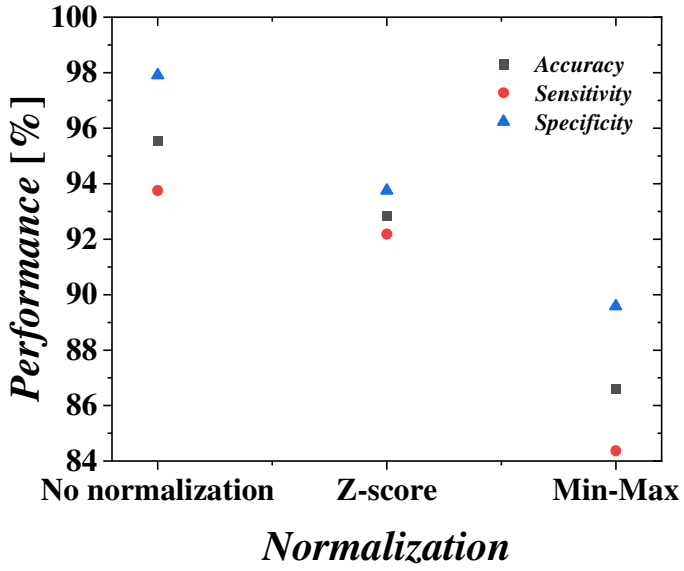


Figure 9. Validation data performances of the proposed STM-LSTM-based fall detection system, applying normalization methods.

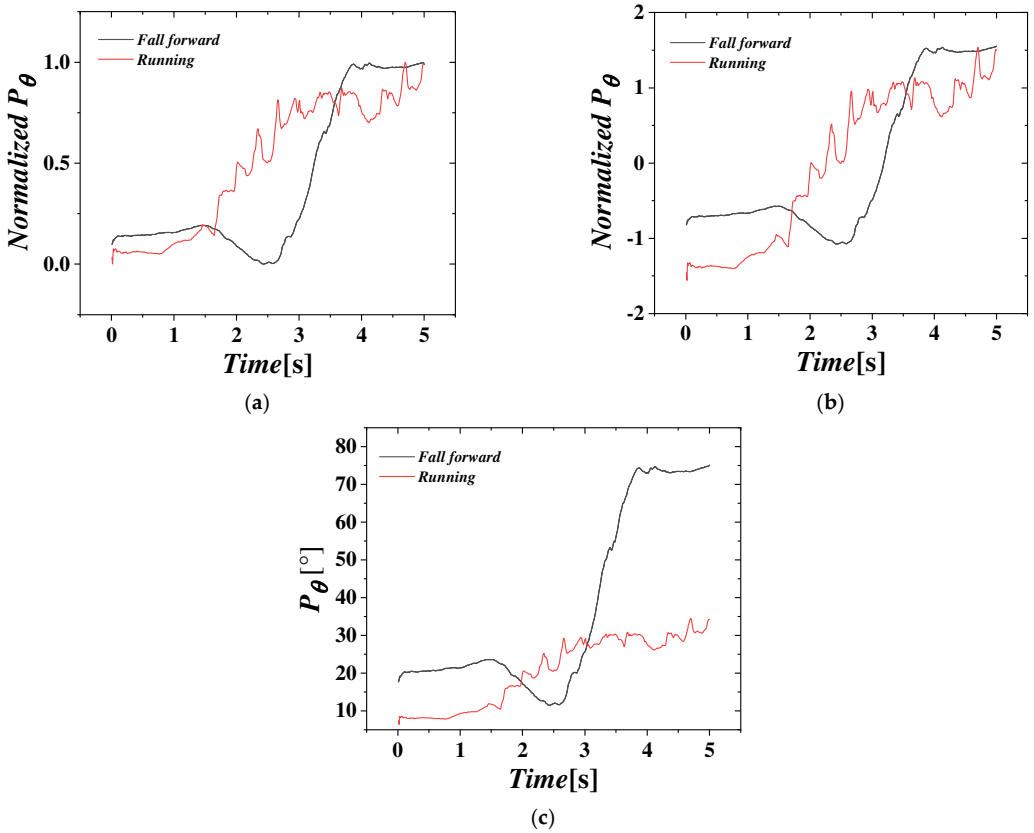


Figure 10. Comparison of data of running and forward fall before and after normalization. (**a**) min-max normalization, (**b**) Z-score normalization, (**c**) No-normalization.

5.2. Sampling and Regularization Rate

In this section, the impact of different samplings of input parameter and different regularization rates in the STM-LSTM-based fall detection system is examined. Figure 11 shows validation data accuracy vs. regularization rate in the STM-LSTM-based fall detection system with respect to the different sampling intervals of input parameter. The regularization rates are 0.00015, 0.00065, 0.0015, 0.0065, 0.015, 0.065, and 0.15. The black squares, red circles, blue up-triangles, green down-triangles, and purple diamonds denote different sampling intervals of 1, 3, 5, 7, and 9, respectively. Peak values of accuracy of sampling intervals 1, 3, 5, 7, and 9 are 96.42% at $\lambda = 0.00065$, 96.42% at $\lambda = 0.0015$, 95.53% at $\lambda = 0.0065$, 94.64% at $\lambda = 0.0065$, and 94.64% at $\lambda = 0.0065$, respectively. As the sampling interval increases to 5, the regularization rate at the peak of accuracy increases, but when it is 5 or more, the regularization rate at the peak of accuracy is the same. When the sampling interval increases, the peak values of accuracy decrease. As the sampling interval increases, the length of input data decreases, and thus the noise can be slightly reduced, and the complexity of the input data can be reduced. Therefore, overfitting can occur due to a small number of samples [47]. An excessive increase of sampling interval is more dependent on the simplification of data than the noise reduction, resulting in a decrease in accuracy than an increase in overfitting.

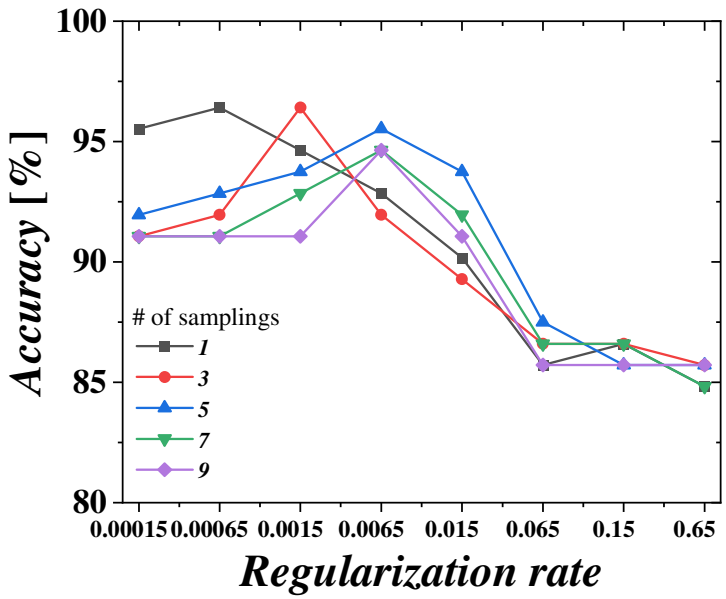


Figure 11. Validation data accuracy vs. regularization rate of the STM-LSTM-based fall detection system with respect to different sampling intervals.

5.3. Hidden Layer Node and Regularization Rate

In this section, the impact of different hidden layer nodes and different regularization rates in the STM-LSTM-based fall detection system is examined. Figure 12 shows the validation data accuracy vs. regularization rate of the STM-LSTM-based fall detection system with respect to the different numbers of hidden layer nodes. The regularization rates are 0.00065, 0.0015, 0.0065, 0.015, 0.065, 0.15, and 0.65. The black squares, red circles, blue up-triangles, green down-triangles, purple diamonds, and gold left-triangles denote different numbers of the hidden layer nodes of 6, 13, 32, 64, 128, and 256, respectively. Peak values of accuracy of the numbers of the hidden layer nodes, 6, 13, 32, 64, 128, and 256 are 94.64% at $\lambda = 0.00015$, 96.42% at $\lambda = 0.00065$, 97.32% at $\lambda = 0.0015$, 97.32% at $\lambda = 0.0065$, 98.21% at $\lambda = 0.015$, and 98.21% at $\lambda = 0.065$, respectively. As the number of the hidden layer nodes increases, the regularization rate at the peak of accuracy increases and the peak values of accuracy increase. This means that as the number of hidden layer nodes increases, both accuracy and overfitting can increase due to the increase of network complexity [26,47], and thus the overfitting can be suppressed by increasing the regularization rate [48].

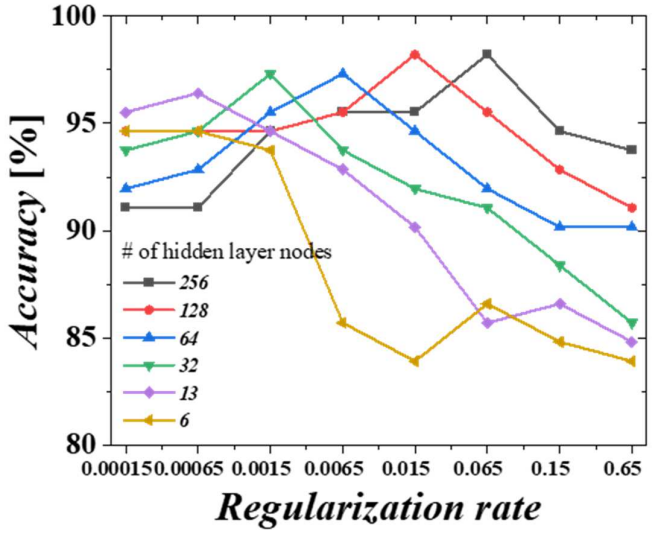


Figure 12. Validation data accuracy vs. regularization rate of the STM-LSTM-based fall detection system with respect to different number of hidden layer nodes.

Meanwhile, as the network complexity increases, the training time also increases, and thus it is necessary to efficiently select the number of hidden layer nodes suitable for both higher accuracy and shorter training time. Figure 13 shows the highest validation data accuracy and computation time of the STM-LSTM-based fall detection system with respect to different number of hidden layer nodes. As the number of hidden layer nodes increases, the highest validation data accuracy increases, and computation time increases rapidly at 256 nodes. It showed the same best accuracy in the number of hidden layer nodes of 128 and 256. Accordingly, it can be determined that the number of hidden layer nodes of 128 is most appropriate in terms of accuracy and time.

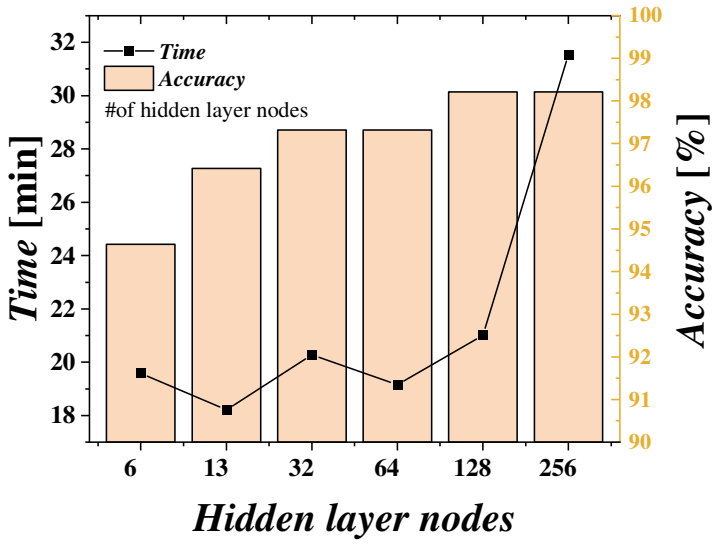


Figure 13. Highest validation data accuracy and computation time of the STM-LSTM-based fall detection system with respect to different number of hidden layer nodes.

5.4. Summary

In Section 4, the training data accuracy of STM-LSTM-based fall detection system showed 100% for the single parameter P_G and the multiple parameter P_{ALL}, and thus it may be overfitted in practice. To resolve the overfitting issue and find optimal values of the parameters used in LSTM, the validation data performance calculated from the validation data, using the network parameters obtained from the training data, was investigated with

respect to normalization method, sampling, regularization rate, and hidden layer node, as follows:

1. As some human activity patterns after the Min-Max or the Z-score normalizations are so similar to each other and are difficult to distinguish, the accuracy of distinguishing falls tends to decrease, and this leads to the conclusion that a normalization is not suitable for this fall detection.
2. When the sampling interval increases, the peak values of accuracy decrease due to overfitting by reduced input data. Therefore, larger sampling intervals require higher regularization to reduce overfitting.
3. As the number of hidden layer nodes increases, both accuracy and overfitting can increase due to the increase of network complexity. Therefore, the higher the hidden layer node and the higher the regularization rate, the higher the accuracy and the lower the overfitting, respectively.
4. In terms of higher accuracy and shorter computation time, the optimal values of the parameters of the LSTM and normalization method are found as follows: No-normalization and no-sampling with 128 hidden layer nodes and regularization rate of 0.015. This is best accuracy of 98.21% with a relatively short computation time.

6. Conclusions

In this paper, the STM-LSTM-based fall detection system that combines the simple threshold method and the LSTM was proposed. The proposed system was based on the single and multiple parameters calculated from the three-axial acceleration data. In training mode, the parameters were normalized and then the parameters were learned using LSTM. In test mode, the first possible fall was primarily determined when both P_S and P_θ exceeded the thresholds of 2.5g and 46.42°, respectively. The first possible fall data were just classified using LSTM learned in training mode. To examine validity of the proposed STM-LSTM fall detection system, it was compared with the previously reported STM-HMM-based fall detection system. The best training accuracy by the STM-LSTM-based fall detection system is 100% for P_G, 0.5% higher than that of the STM-HMM-based system. However, since training data accuracy for this comparison is a result of training data only without validity data, the risk of overfitting may be occurred. To solve the overfitting problem and find optimal values of the parameters used in LSTM, the validation data performance calculated from the validation data, using the network parameters obtained from the training data, was investigated with respect to normalization method, sampling, regularization rate, and hidden layer node. For normalization and sampling, it showed the best performance in no normalization and no sampling. The computation time by 128 hidden layer nodes was significantly shorter than that of 256 nodes although the same best accuracy was 98.21% for both 128 and 256 nodes. As the number of hidden layer nodes or sampling interval increase, the regularization rate at the highest value of accuracy increases. This can be interpreted as suppressing overfitting by increasing the regularization rate. Therefore, the fall detection efficiency can be improved by selecting an appropriate number of hidden layer nodes and regularization rate.

However, this paper still has many directions worthy of improvement. The proposed STM-LSTM fall detection system was compared with the previously reported STM-HMM fall detection system, and the data set of the previously reported STM-HMM fall detection system was used and it has the following limitations: First, it is necessary to verify a reliability through comparison with the data set acquired from the self-developed embedded edge device and that from commercial device. Second, it is very simple because it consists only of four types of ADL and three types of fall gestures. Third, as the subject population consists of four men and two women, it has a gender and age imbalance. To solve this dataset problem, it is necessary to apply this proposed STM-LSTM fall detection system to public datasets verified for fall detection. When public datasets are applied to the proposed system in the future, they are pre-processed with single, double, and multiple parameters,

and optimal network parameters can be newly and easily found using the optimal network parameters and optimization method introduced in this paper.

Author Contributions: Conceptualization, S.S.J., N.H.K., and Y.S.Y.; methodology, S.S.J., N.H.K., and Y.S.Y.; investigation, S.S.J. and Y.S.Y.; data curation, S.S.J.; writing—original draft preparation, S.S.J. and Y.S.Y.; writing—review and editing, S.S.J. and Y.S.Y.; supervision, Y.S.Y.; project administration, Y.S.Y.; funding acquisition, Y.S.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Basic Science Research Program through NRF of Korea funded by the Ministry of Education (NRF-2019R1F1A1060383).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alshammari, S.A.; Alhassan, A.M.; Aldawsari, M.A.; Bazuhair, F.O.; Alotaibi, F.K.; Aldakhil, A.A.; Abdulfattah, F.W. Falls among elderly and its relation with their health problems and surrounding environmental factors in Riyadh. *J. Fam. Community Med.* **2018**, *25*, 29–34. [CrossRef]
2. World Health Organization. Available online: https://www.who.int/zh/news-room/fact-sheets/detail/falls (accessed on 9 March 2020).
3. National Health Administration, Ministry of Health and Welfare. Available online: https://www.hpa.gov.tw/Pages/Detail.aspx?nodeid=807&pid=4326 (accessed on 9 March 2020).
4. Igual, R.; Medrano, C.; Plaza, I. Challenges, issues and trends in fall detection systems. *BioMed. Eng. Online* **2013**, *12*, 66. [CrossRef]
5. Wang, X.; Ellul, J.; Azzopardi, G. Elderly fall detection systems: A literature survey. *Frontiers* **2020**, *7*, 71. [CrossRef]
6. Ramachandran, A.; Karuppiah, A. A survey on recent advances in wearable fall detection systems. *BioMed Res. Int.* **2020**, *2020*, 2167160. [CrossRef]
7. Ren, L.; Peng, Y. Research of fall detection and fall prevention technologies: A systematic review. *IEEE Access* **2019**, *7*, 77702–77722. [CrossRef]
8. Taylor, R.M.; Marc, E.C.; Vangeli, S.M.; Anne, H.H.N.; Coralys, C.R. SmartFall: A smartwatch-based fall detection system using deep learning. *Sensors* **2018**, *18*, 10. [CrossRef]
9. Vilarinho, T.; Farshchian, B.; Bajer, D.G.; Dahl, O.H.; Egge, I.; Hegdal, S.S.; Lones, A.; Slettevold, J.N. A combined smartphone and smartwatch fall detection system. In Proceedings of the 2015 IEEE International Conference on Computer and Information Technology, Ubiquitous Computing and Communications, Dependable, Autonomic and Secure Computing, Pervasive Intelligence and Computing, Liverpool, UK, 26–28 October 2015; pp. 1443–1448. [CrossRef]
10. Casilari, E.; Oviedo-Jiménez, M.A. Automatic fall detection system based on the combined use of a smartphone and a smartwatch. *PLoS ONE* **2015**, *10*, 11. [CrossRef]
11. Habib, M.A.; Mohktar, M.S.; Kamaruzzaman, S.B.; Lim, K.S.; Pin, T.M.; Ibrahim, F. Smartphone-based solutions for fall detection and prevention: Challenges and open issues. *Sensors* **2014**, *14*, 4. [CrossRef]
12. Rakhman, A.Z.; Nugroho, L.E. Fall detection system using accelerometer and gyroscope based on smartphone. In Proceedings of the 1st International Conference on Information Technology, Computer, and Electrical Engineering, Semarang, Indonesia, 8 November 2014; pp. 99–104. [CrossRef]
13. Yavu, G.; Kocak, M.; Ergun, G.; Alemdar, H.O.; Yalcin, H.; Incel, O.D.; Ersoy, C. A smartphone based fall detector with online location support. In Proceedings of the International Workshop on Sensing for App Phones (ACM), Zurich, Switzerland, 2 November 2010; pp. 31–35.
14. He, Y.; Li, Y.; Bao, S.D. Fall detection by built-in tri-accelerometer of smartphone. In Proceedings of the 2012 IEEE-EMBS International Conference on Biomedical and Health Informatics, Hong Kong, China, 5–7 January 2012; pp. 184–187. [CrossRef]
15. Stefano, A.; Marco, A.; Francesco, B.; Guglielmo, C.; Paolo, C.; Alessio, V. A smartphone-based fall detection system. *Pervasive Mob. Comput.* **2012**, *8*, 6. [CrossRef]
16. Yi, Y.J.; Yu, Y.S. Emergency-monitoring system based on newly-developed fall detection algorithm. *J. Inf. Commun. Converg. Eng.* **2013**, *11*, 3. [CrossRef]
17. Lim, D.H.; Park, C.H.; Kim, N.H. Fall-detection algorithm using 3-axis acceleration: Combination with simple threshold and hidden Markov model. *J. Appl. Math.* **2014**, *2014*, 896030. [CrossRef]
18. Jiang, M.; Chen, Y.; Zhao, Y.; Cai, A. A real-time fall detection system based on HMM and RVM. In Proceedings of the 2013 Visual Communications and Image Processing (VCIP), Kuching, Malaysia, 17–20 November 2013; pp. 1–6. [CrossRef]

19. Thome, N.; Miguet, S.; Ambellouis, S. A real-time, multiview fall detection system: A LHMM-based approach. *IEEE Trans. Circuits Syst. Video Technol.* **2008**, *18*, 1522–1532. [CrossRef]
20. Mistikoglu, G.; Gerek, I.H.; Erdis, E.; Usmen, P.M.; Cakan, H.; Kazan, E.E. Decision tree analysis of construction fall accidents involving roofers. *Expert Syst. Appl.* **2015**, *42*, 2256–2263. [CrossRef]
21. Liu, C.L.; Lee, C.H.; Lin, P.M. A fall detection system using k-nearest neighbor classifier. *Expert Syst. Appl.* **2010**, *37*, 7174–7181. [CrossRef]
22. Liu, L.; Popescu, M.; Skubic, M.; Rantz, M.; Yardibi, T.; Cuddihy, P. Automatic fall detection based on Doppler radar motion signature. In Proceedings of the 2011 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth), Dublin, Ireland, 23–26 May 2011; pp. 222–225. [CrossRef]
23. Zhang, T.; Wang, J.; Xu, L.; Liu, P. Fall Detection by Wearable Sensor and One-Class SVM Algorithm. In *Intelligent Computing in Signal Processing and Pattern Recognition*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 858–863. [CrossRef]
24. Ghahramani, Z. An introduction to hidden Markov models and Bayesian networks. *Int. J. Patt. Recogn. Artif. Intell.* **2001**, *15*, 9–42. [CrossRef]
25. Anguita, D.; Ghio, A.; Greco, N.; Oneto, L.; Ridella, S. Model selection for support vector machines: Advantages and disadvantages of the machine learning theory. In Proceedings of the 2010 International Joint Conference on Neural Networks (IJNN), Barcelona, Spain, 18–23 July 2010; pp. 1–8. [CrossRef]
26. Lin, C.B.; Dong, Z.; Kuan, W.K.; Huang, Y.F. A framework for fall detection based on OpenPose skeleton and LSTM/GRU models. *Appl. Sci.* **2020**, *11*, 1. [CrossRef]
27. Chen, W.; Jiang, Z.; Guo, H.; Ni, X. Fall detection based on key points of human-skeleton using openpose. *Symmetry* **2020**, *12*, 744. [CrossRef]
28. Ajerla, D.; Mahfuz, S.; Zulkernine, F. A real-time patient monitoring framework for fall detection. *Wirel. Commun. Mob. Comput.* **2019**, *2019*, 9507938. [CrossRef]
29. Queralta, J.P.; Gia, T.N.; Tenhunen, H.; Westerlund, T. Edge-AI in LoRa-based health monitoring: Fall detection system with fog computing and LSTM recurrent neural networks. In Proceedings of the 42nd International Conference on Telecommunications and Signal Processing (TSP), IEEE, Budapest, Hungary, 1–3 July 2019; pp. 601–604.
30. Lu, N.; Wu, Y.; Feng, L.; Song, J. Deep learning for fall detection: Three-dimensional CNN combined with LSTM on video kinematic data. *IEEE J. Biomed. Health Inform.* **2018**, *23*, 314–323. [CrossRef]
31. Santos, G.L.; Endo, P.T.; Monteiro, K.H.D.C.; Rocha, E.D.S.; Silva, I.; Lynn, T. Accelerometer-based human fall detection using convolutional neural networks. *Sensors* **2019**, *19*, 7. [CrossRef]
32. Kwolek, B.; Kepski, M. Improving fall detection by the use of depth sensor and accelerometer. *Neurocomputing* **2015**, *168*, 637–645. [CrossRef]
33. Gasparrini, S.; Cippitelli, E.; Spinsante, S.; Gambi, E. A depth-based fall detection system using a Kinect® sensor. *Sensors* **2014**, *14*, 2756–2775. [CrossRef] [PubMed]
34. Maitre, J.; Bouchard, K.; Gaboury, S. Fall detection with UWB radars and CNN-LSTM architecture. *IEEE J. Biomed. Health Inform.* **2020**, *25*, 1273–1283. [CrossRef] [PubMed]
35. Galvão, Y.M.; Ferreira, J.; Albuquerque, V.A.; Barros, P.; Fernandes, B.J. A multimodal approach using deep learning for fall detection. *Expert Syst. Appl.* **2021**, *168*, 114226. [CrossRef]
36. Adhikari, K.; Bouchachia, H.; Nait-Charif, H. Activity recognition for indoor fall detection using convolutional neural network. In Proceedings of the 2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA), Nagoya, Japan, 8–12 May 2017; pp. 81–84. [CrossRef]
37. Casilari, E.; Lora-Rivera, R.; García-Lagos, F. A study on the application of convolutional neural networks to fall detection evaluated with multiple public datasets. *Sensors* **2020**, *20*, 1466. [CrossRef] [PubMed]
38. Hochreiter, S. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *Int. J. Uncertain. Fuzziness Knowl. Based Syst.* **1998**, *6*, 107–116. [CrossRef]
39. Kim, N.H.; Yu, Y.S. Fall recognition algorithm using gravity-weighted 3-axis accelerometer data. *J. Inst. Electron. Eng. Korea* **2012**, *50*, 1570–1575. [CrossRef]
40. Sepp, H.; Jürgen, S. Long short-term memory. *Neural Comput.* **1997**, *9*, 8. [CrossRef]
41. Zhang, Z.; Sabuncu, M. Generalized cross entropy loss for training deep neural networks with noisy labels. In Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada, 3–8 December 2018; pp. 8792–8802. [CrossRef]
42. Rusiecki, A. Trimmed categorical cross-entropy for deep learning with label noise. *Electron. Lett.* **2019**, *55*, 319–320. [CrossRef]
43. Burden, F.; Winkler, D. Bayesian regularization of neural networks. *Artif. Neural Netw.* **2008**, *458*, 23–42. [CrossRef]
44. Park, P.; Marco, P.D.; Shin, H.; Bang, J. Fault detection and diagnosis using combined autoencoder and long short-term memory network. *Sensors* **2019**, *19*, 4612. [CrossRef] [PubMed]
45. Li, K.; Zhao, X.; Bian, J.; Tan, M. Sequential learning for multimodal 3D human activity recognition with long-short term memory. In Proceedings of the 2017 IEEE International Conference on Mechatronics and Automation (ICMA), Takamatsu, Japan, 6–9 August 2017; pp. 1556–1561. [CrossRef]
46. Parikh, R.; Mathai, A.; Parikh, S.; Sekhar, G.C.; Thomas, R. Understanding and using sensitivity, specificity and predictive values. *Indian J. Ophthalmol.* **2008**, *56*, 45. [CrossRef] [PubMed]

47. Bejani, M.M.; Ghatee, M. A systematic review on overfitting control in shallow and deep neural networks. *Artif. Intell. Rev.* **2021**, *54*, 6391–6438. [CrossRef]
48. Wang, G.; Lee, K.-C.; Shin, S.-Y. Novel image classification method based on few-shot learning in monkey species. *J. Inf. Commun. Converg. Eng.* **2021**, *19*, 79–83. [CrossRef]