

Fall Detection System Using Accelerometer and Gyroscope Sensors with Backpropagation Neural Networks

Helmy Fitriawan^{1*}, Alby Dzaky Hadi Pamungkas², Ezza Ahmad Faturrohman³,
Sri Purwiyanti⁴, Anisa Ulya Darajat⁵, Teddy Surya Gunawan⁶

Dept. of Electrical Engineering, University of Lampung, 35145 Bandar Lampung, Indonesia^{1, 2, 3, 4, 5}
Electrical and Computer Engineering Dept., International Islamic University Malaysia, 53100 Kuala Lumpur, Malaysia⁶

Abstract—A fall is an incident in which a person suddenly loses their balance and falls uncontrollably to the ground or another surface. This is particularly dangerous for the elderly, as it can lead to serious injury or even death if not treated immediately. This study suggests utilizing a Backpropagation Neural Network (BPNN) to classify fall and non-fall incidents, particularly in older adults, by measuring alterations in linear acceleration and angular velocity obtained from accelerometer and gyroscope sensors. A dataset featuring eight distinct movement types, including standing to bowing, standing to squatting, sitting to standing, standing to prostrating, sitting to prostrating, falling sideways, falling forward, and falling backward, is collected from 69 participants and processed by determining the difference (Δ) between consecutive sensor measurements, followed by normalization, labeling, and statistical feature extraction. The collected features are fed into a BPNN that has two hidden layers, each with 64 and 128 neurons. The proposed model achieved an accuracy of 86.87%, validation accuracy of 85.00%, and testing accuracy of 92.86%. Furthermore, test results show a precision of 94%, recall of 85%, and specificity of 97%, demonstrating the system's ability to accurately classify eight different human activities and distinguish falls from normal daily activities. These findings indicate that a Backpropagation Artificial Neural Network in conjunction with the suggested Δ -based preprocessing offers an efficient method for inertial sensor-based fall detection systems.

Keywords—Backpropagation neural network; fall detection; human activity recognition; accelerometer; gyroscope

I. INTRODUCTION

Patient safety is of utmost importance, and falls are among the most frequent incidents affecting elderly adults. For elderly patients, experiencing unnoticed falls can pose a hidden danger. The immediate consequences of a fall, such as injury, can be worse if assistance is not obtained immediately. Statistics indicate that the most serious outcomes result from delays in receiving help and medical care, rather than from the fall itself [1-4].

In the healthcare sector, the development of fall detection devices has become a prominent research topic. Recent advancements in Micro-Electro-Mechanical Systems (MEMS) sensor and actuator technology have enabled the development of fall detection systems using accelerometers and gyroscopes [5-8]. This technique relies on detecting and tracking changes in

acceleration along three orthogonal directions to identify changes in a person's body position and movement. A threshold-based fall detection system uses wearable sensors to detect sudden body movements. The system monitors when total acceleration exceeds a pre-set numerical threshold value. If the data surpasses that limit, the system flags a potential fall event [9-12]. The collected data is continuously analyzed algorithmically to determine whether the individual has fallen. If a fall is detected, the device can use GPS and a wireless transmitter to determine the location and issue an alert for help. Research has also been conducted on fall detection using smartphone accelerometers, where the system can send an SMS containing an alert message indicating that the patient has fallen, along with the person's coordinates, and automatically connect the call to a pre-registered number [13-16]. The core elements of fall detection are reliable detection principles and practical algorithms for assessing the existence of emergencies.

Various classification methods have been applied to improve the accuracy of fall detection. Research using the Fuzzy Logic method has been able to detect falls based on a rule base constructed from changes in acceleration values and body orientation. However, the performance of this method is highly dependent on the formulation of rules and membership functions, making it less flexible in recognizing complex movement variations [17-20]. Another study using the K-Nearest Neighbor (KNN) algorithm showed that accelerometer and gyroscope data can be classified with a good level of accuracy, but its performance is influenced by the choice of the K value, the distribution of the training data, and increased computation time as the dataset size grows [21-23].

Addressing these issues, this study develops an activity classification and fall detection model using a Backpropagation Neural Network that utilizes accelerometer and gyroscope data from the MPU6050 sensor. This study employs a relatively large dataset—comprising 36,432 data points collected from 69 participants—covering eight types of movements: standing to bowing, standing to squatting, sitting to standing, standing to prostrating, sitting to prostrating, falling sideways, falling forward, and falling backward. Furthermore, the study implements preprocessing stages—including the calculation of inter-sample value changes (Δ), normalization, and data labeling—prior to the training process to generate more representative features. The primary contribution of this

*Corresponding author.
Higher Education for Technology and Innovation (HETI) Project.

research is the development of a Backpropagation Neural Network model that utilizes a larger dataset and comprehensive preprocessing steps to enhance the classification performance for these eight movement types based on MPU6050 sensor data. Model performance is evaluated using a confusion matrix based on accuracy, precision, recall, and specificity metrics to assess the ability of the model to accurately recognize activities and detect falls [24-26].

The following sections of this study are organized as follows: Section II presents the stages of this research and proposes the backpropagation neural network architecture employed. Model training and the discussion of test results are presented in Section III. Finally, research findings are concluded in Section IV.

II. METHOD

This study aims to develop a fall detection system utilizing accelerometer and gyroscope sensors integrated into an MPU6050 module. An Artificial Neural Network (ANN) employing the backpropagation method is used to process sensor data and identify various movement patterns, thereby distinguishing between falls and normal daily activities.

Fig. 1 shows the workflow of the research methodology. In the initial phase of the research, the MPU6050 module, comprising accelerometer and gyroscope sensors mounted on a wearable vest, is used to collect data. These sensors continuously recorded movement data from subjects performing various movement scenarios, including simulated falls and daily activities. The collected raw signals subsequently underwent preprocessing to remove noise and normalize the data, ensuring uniformity for further analysis.

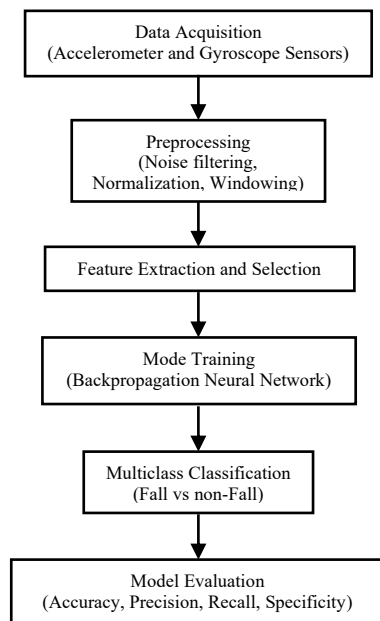


Fig. 1. Research stages of a fall detection system using a Backpropagation Neural Network (BPNN).

Following preprocessing, pertinent movement characteristics that characterize various movement patterns are obtained using a feature extraction and selection process. A BPNN model is then trained using the backpropagation algorithm utilizing the chosen features as input. The data set is split into training and testing subsets in order to assess the model performance and guarantee cross-validation of the classification procedure. The BPNN architecture is optimized by varying the number of hidden layers and neurons to achieve the highest accuracy. Model performance is evaluated based on accuracy, precision, recall, and specificity.

The following part provides a detailed description of how the proposed fall detection system is implemented.

A. Data Collection

The dataset in this study is collected using primary data acquisition from subjects wearing a vest equipped with the MPU6050 sensor module, which combines a 3-axis accelerometer and a 3-axis gyroscope. This sensor is placed on the user's shoulder and operated via an Arduino-based system. This placement is chosen to capture movement data and changes in body orientation with greater accuracy, given that the shoulder area is relatively stable and less prone to the extraneous movements often associated with the arms or legs. Thanks to this consistent and stable positioning, the sensor can measure angular changes and body movements with a higher degree of precision. Furthermore, the shoulder surface, which tends to remain relatively parallel to the earth, helps enhance measurement accuracy while minimizing interference caused by unexpected movements.

The dataset consists of motion signals recorded from simulated fall events as well as normal daily activities. In this research, the dataset was collected from 69 participants, where data from 62 participants are used for model training and data from the remaining 7 participants are used for model testing. Each participant took part in a data collection session covering two types of movements: non-fall movements (standing to bowing, standing to squatting, sitting to standing, standing to prostrating, and sitting to prostrating) and fall movements (falling sideways, falling forward, and falling backward). Conducting fall experiments on older adults entails certain risks, even under controlled settings; therefore, the fall simulations for this study were conducted exclusively and in a controlled manner on young participants between the ages of 20 and 24.

In order to gain coherent movement patterns, each activity is recorded from the initial position to the final position of the movement. Specifically for simulated fall activities, the recording process begins when the subject is in an upright position, continues as the subject loses balance, and concludes when the subject reaches the final position after the fall. This series of stages is carried out to fully represent the fall movement pattern, which will subsequently be used as input data for the BPNN. An illustration of the data acquisition process when a participant experiences a fall is shown in Fig. 2. The complete dataset is then split into training and testing subsets to evaluate the model performance.



Fig. 2. Illustration of data collection from a subject who has experienced a fall.

B. Preprocessing and Feature Selection

Raw data obtained from sensors undergoes a preprocessing stage to enhance its quality and consistency. These preprocessing steps include noise filtering, normalization, and the segmentation of time-series signals into windows suitable for feature extraction. This stage ensures uniform data scaling and minimizes measurement errors, rendering the data more suitable for analysis using an ANN. Data is measured and recorded using an MPU6050 sensor, which yields six parameters: acceleration (A_x , A_y , and A_z) and angular velocity (G_x , G_y , and G_z) along the x, y, and z axes, respectively. Feature extraction and selection are carried out to identify motion characteristics most relevant to fall detection. Each activity is recorded over 11 samples, so that each movement produced an 11 x 6 data matrix representing changes in acceleration and body orientation throughout the activity, as presented in Table I.

TABLE I. RAW DATA FROM ACCELEROMETER AND GYROSCOPE SENSOR READINGS BEFORE PREPROCESSING

No	Accelerometer (A)			Gyroscope (G)		
	A_x	A_y	A_z	G_x	G_y	G_z
1	-1.60	-3.70	8.52	0.09	0.09	-0.24
2	-1.66	-3.71	8.51	0.64	0.49	-1.19
3	-1.53	-3.76	8.49	0.99	0.05	-4.84
4	-1.47	-3.70	8.53	-0.17	-0.40	0.40
5	-1.53	-3.71	8.53	-1.59	-3.97	-0.96
6	-1.79	-3.60	7.29	-19.73	-57.05	-12.08
7	6.93	-4.36	3.65	-24.90	-103.53	-35.62
8	9.94	-4.41	-1.97	-5.69	-53.59	-24.87
9	9.79	-3.29	-3.38	0	-0.95	-7.31
10	8.75	-3.52	-2.78	4.18	-0.12	-1.37
11	9.05	-3.32	-2.91	1.98	-0.98	-1.34

Before being used as input for the BPNN model, the data undergoes preprocessing steps that include dataset preparation, calculation of the change in values between samples (Δ), normalization, and data labeling. After the raw data is obtained, the next step is to calculate the difference (Δ) between samples for each parameter of the accelerometer (A_x , A_y , and A_z) and the gyroscope (G_x , G_y , and G_z). This calculation aims to represent the changes in acceleration and angular velocity that occur during the activity, so that movement characteristics can be captured more accurately than when using raw sensor values. Additionally, using the Δ values helps reduce the influence of

differences in the initial sensor positions among participants. Since each activity consists of 11 samples, the Δ calculation process yields 10 data points for each sensor parameter, which are then used as input features in the Backpropagation Neural Network model as presented in Table II.

Based on the calculated change values (Δ), each data point is then labelled according to the type of activity performed. The labelling process aims to distinguish eight movement classes: standing to bowing, standing to squatting, sitting to standing, standing to prostrating, sitting to prostrating, falling sideways, falling forward, and falling backward. The data that has undergone the change value calculation and labelling processes is then used as input for the feature extraction stage and the training of the BPNN.

TABLE II. SAMPLE OF DIFFERENCE (Δ) VALUES OBTAINED FROM ACCELEROMETER AND GYROSCOPE DATA

No	Accelerometer (A)			Gyroscope (G)		
	ΔA_x	ΔA_y	ΔA_z	ΔG_x	ΔG_y	ΔG_z
1	-0.06	-0.01	-0.01	0.55	0.40	-0.95
2	0.13	-0.05	-0.02	0.35	-0.44	-3.65
3	0.06	0.06	0.04	-1.16	-0.45	5.24
4	-0.06	-0.01	0	-1.42	-3.57	-1.36
5	-0.26	0.11	-1.24	-18.14	-53.08	-11.12
6	8.72	-0.76	-3.64	-5.17	-46.48	-23.54
7	3.01	-0.05	-5.62	19.21	49.94	10.75
8	-0.15	1.12	-1.41	5.69	52.64	17.56
9	-1.04	-0.23	0.60	4.18	0.83	5.94
10	0.30	0.20	-0.13	-2.20	-0.86	0.03

C. Modelling

In the modelling phase, the classification process is carried out using an ANN trained with the backpropagation algorithm. The model training process is implemented using the Python programming language, which offers flexibility in optimizing network parameters. Before the training process begins, each parameter representing the change (Δ) is transformed into statistical features consisting of the minimum (Min), maximum (Max), mean (Mean), and standard deviation (Standard Deviation) values [27-29]. This feature extraction process summarizes the key characteristics of each activity, reducing data complexity while preserving essential information that represents movement patterns. The minimum and maximum values represent the lowest and highest sensor readings during the activity, respectively; the mean value describes the trend of sensor changes, while the standard deviation measures data variation relative to that mean. These four statistical features are calculated for each accelerometer axis (A_x , A_y , A_z) and gyroscope axis (G_x , G_y , G_z), resulting in a total of 24 input features.

The extracted features are then used to train the Backpropagation Neural Network. The neural network architecture consists of one input layer, two hidden layers, and one output layer. The hidden layers process the extracted features, while the output layer generates binary or multi-class

classification results depending on the scenario. The architecture of the proposed BPNN model employed in this study is illustrated in Fig. 3. The input layer contains 24 neurons ($i_1, i_2, \dots, i_{24}$) corresponding to the extracted statistical features. The first and second hidden layers consist of 128 ($h_{1,1}, h_{1,2}, \dots, h_{1,128}$) and 64 ($h_{2,1}, h_{2,2}, \dots, h_{2,64}$) neurons, respectively, while the output layer consists of 8 neurons ($o_1, o_2, \dots, o_8$) representing the eight observed activity classes.

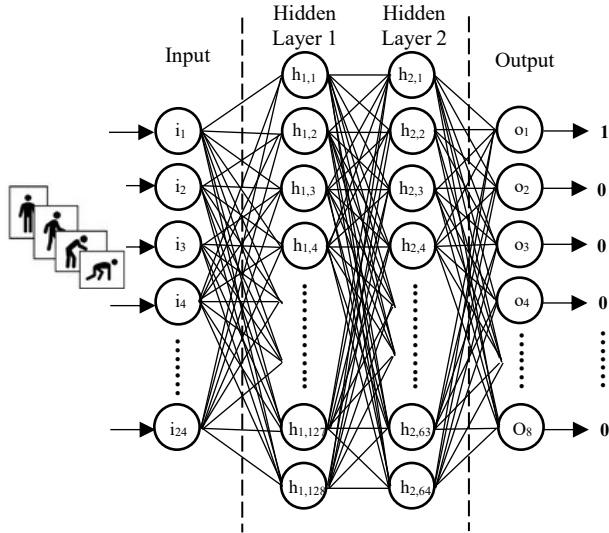


Fig. 3. Architecture of the backpropagation neural network employed in this study.

The input layer receives 24 statistical features derived from accelerometer and gyroscope data obtained during the pre-processing and feature selection stages. The output layer has 8 neurons representing eight activity classes: class 0 for standing to bowing, class 1 for standing to squatting, class 2 for sitting to standing, class 3 for standing to prostrating, class 4 for sitting to prostrating, class 5 for falling sideways, class 6 for falling forward, and class 7 for falling backward.

The proposed network incorporates two hidden layers because the accelerometer and gyroscope data exhibit complex non-linear characteristics. A narrowing structure (from 128 neurons to 64 neurons) was selected for this network to mimic the principle of gradual feature abstraction. The first hidden layer (128 neurons) serves to learn basic patterns and raw features from the input, while the second hidden layer (64 neurons) captures more complex relationships between features and distills that information into a more concise and meaningful representation, thereby enabling more optimal classification at the output layer [30]. Research in [31] indicates that the 128-64 neuron configuration achieves an optimal point (sweet spot) on the performance curve. This configuration yields high accuracy without showing signs of overfitting, thereby maintaining the model's flexibility when processing new data.

Both hidden layers use the ReLU activation function to handle non-linear decision boundaries. These functions satisfy the properties of continuity, differentiability, and monotonicity, which are essential for the learning process in artificial neural networks. It also accelerates the convergence during the network training process and prevents the vanishing gradient problem

often encountered in artificial neural network models [32-33]. The ReLU (Rectified Linear Unit) activation function transforms negative input values into zero and passes positive values through directly, using Eq. (1).

$$ReLU(z) = \max(0, z) \quad (1)$$

where, z is the neuron input value; if the input value is below zero ($z < 0$), the output is 0, and if the input value is greater than or equal to zero ($z \geq 0$), the output remains equal to the input value itself.

The output layer employs the Softmax activation function to transform the network channels into a probability distribution across the eight activity classes. Each probability value lies between 0 and 1, and the total sum of values at the output layer is always exactly 1 [34-35]. The Softmax function is defined in Eq. (2).

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{j=1}^n e^{z_j}} \quad (2)$$

where, $\hat{y}_i$ is the probability of the i^{th} class or the actual true label, z is output of the neuron before activation, e^{z_i} is the exponential function, n is the number of output classes, and $\sum_{j=1}^n e^{z_j}$ is the sum of all exponential values. The class with the highest probability value is the network prediction.

The training phase utilizes a training dataset and involves forward propagation, loss calculation, and bias and weight updates using the backpropagation algorithm. This process yields accuracy and loss metrics that indicate the quality of the training. The final weights and biases obtained from the training process are subsequently used to classify activities based on the testing dataset.

Forward propagation processes input data in a single forward direction through the layers of a neural network to generate a final prediction. Each neuron within a layer processes incoming values, x , using specific weights, w , and biases, b , transforms them via an activation function, and then passes the result, z , to the next layer, as shown in Eq. (3).

$$z = x \cdot w + b \quad (3)$$

After the forward propagation process generates a prediction at the output layer, the loss value is calculated to measure the magnitude of the difference between the network prediction and the actual ground-truth label. In our proposed BPNN architecture, the categorical cross-entropy loss function is used as it is suitable for multiclass classification problems, specifically the eight-class classification as discussed here. It measures the dissimilarity between the predicted probability distribution and the true one-hot encoded label distribution. The lower the resulting loss value, the better the model's ability to predict the correct class. The equation for the categorical cross-entropy loss function is shown in Eq. (4).

$$L = - \sum_i^n y_i \log(\hat{y}_i) \quad (4)$$

Once the loss value is obtained, the backpropagation process is performed to calculate the error gradients with respect to each weight and bias in the network. These gradients are used to iteratively update the network parameters so that the loss value decreases and the model's ability to classify activities improves.

Eq. (5) shows the weight update in the backpropagation algorithm using the network learning rate, denoted by η .

$$W_{new} = W_{old} - \eta \cdot \frac{dL}{dW} \quad (5)$$

A similar equation is also applied to the update in bias values.

D. Evaluation

The evaluation of the proposed fall detection system is conducted using a confusion matrix, which provides a detailed representation of classification results by distinguishing between correctly and incorrectly predicted classes. A multiclass confusion matrix for a neural network is an $N \times N$ grid (where N is the number of classes) that contrasts the network-predicted labels (columns) with the true target labels (rows). The diagonal cells display accurate predictions, while off-diagonal cells reveal particular misclassifications. The confusion matrix consists of four components: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN). These values enable the calculation of various performance metrics, including accuracy, precision, recall, and specificity, which are determined using the following equations.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (6)$$

$$Precision = \frac{TP}{TP+FP} \quad (7)$$

$$Recall = \frac{TP}{TP+FN} \quad (8)$$

$$Specificity = \frac{TN}{TN+FP} \quad (9)$$

Accuracy is the total correct predictions (sum of the diagonal) divided by the total samples; precision is true positives divided by all positive predictions, recall is true positives divided by the total actual positives; and specificity measures the proportion of actual negative cases correctly identified as negative by the neural network.

From the optimization perspective, the loss function guides network training, while these evaluation metrics evaluate real-world practical performance. In this study context, they provide a thorough review of an ANN-based fall detection system while guaranteeing classification accuracy and dependability.

III. RESULTS AND DISCUSSION

This section presents the results of the proposed BPNN training process using the preprocessed training dataset, as well as the testing results using the test dataset. Afterward, the performance of the model is evaluated using metrics derived from the confusion matrix.

A. Model Training

A neural network learns patterns from data by modifying its internal biases and weights during training. The training process is evaluated based on changes in accuracy and loss values at each epoch to assess the model ability to learn movement patterns and its convergence rate. The variation in accuracy values throughout the training process is depicted in Fig. 4. At the beginning of the training, the accuracy level is low because the model was not yet able to optimally recognize data patterns.

As the number of epochs increased, accuracy improved significantly, eventually reaching approximately 86% by the end of the training. The curve, which begins to flatten in subsequent epochs, indicates that the model has achieved convergence and is capable of effectively learning the characteristics of the training dataset of movement patterns.

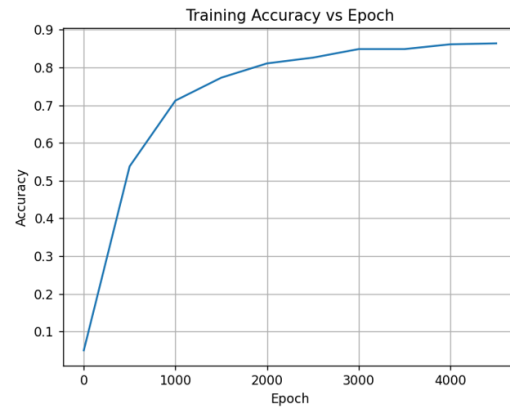


Fig. 4. Plot of accuracy vs epoch for training data.

The loss value indicates the magnitude of the prediction error produced by the model during the training process. The smaller the loss value, the better the model's ability to learn data patterns. Fig. 5 shows that the loss value decreases significantly at the beginning of the training process. As the number of epochs increases, the loss value continues to decrease until it reaches a relatively stable state. This decrease indicates that the optimization process has successfully minimized prediction errors and the model has reached a state of convergence.

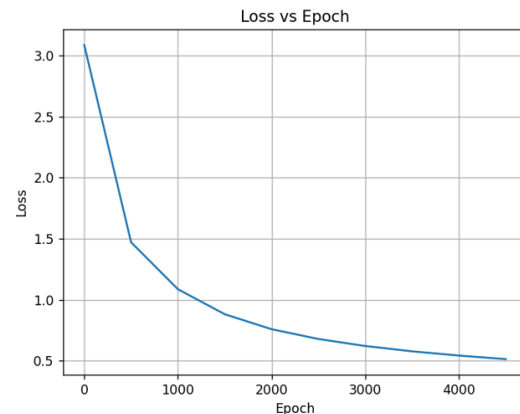


Fig. 5. Plot of loss vs epoch for training data.

B. Classification Performance

After the training process is completed, the proposed BPNN model is evaluated using test data to determine its ability to classify eight types of activities based on accelerometer and gyroscope sensor data. The evaluation is conducted using a confusion matrix, which provides information on the number of correct and incorrect predictions for each activity class. Based on this confusion matrix, the accuracy, precision, recall, and specificity are then calculated to measure the model's overall performance, as shown in Fig. 6.

	Movement Pattern	Prediction							
		Standing to Bowing (0)	Standing to Squatting (1)	Sitting to Standing (2)	Standing to Prostrating (3)	Sitting to Prostrating (4)	Falling Sideways (5)	Falling Forward (6)	Falling Backward (7)
Actual Data	Standing to Bowing (0)	7	0	0	0	0	0	0	0
	Standing to Squatting (1)	1	6	0	0	0	0	0	0
	Sitting to Standing (2)	0	0	7	0	0	0	0	0
	Standing to Prostrating (3)	0	0	0	7	0	0	0	0
	Sitting to Prostrating (4)	0	0	0	0	7	0	0	0
	Falling Sideways (5)	0	0	0	0	0	6	0	1
	Falling Forward (6)	0	0	0	2	0	0	5	0
	Falling Backward (7)	0	0	0	0	0	0	0	7

Fig. 6. Confusion matrix of the BPNN to classify eight different movement patterns, including fall and non-fall.

The model is tested using 56 test data points consisting of seven participants across eight movement pattern classes. The test results show that 52 data points are classified correctly, while 4 data points are misclassified. These errors consisted of one sample of standing to squatting that is predicted as standing to bowing, two samples of falling forward that are predicted as standing to prostrating, and one sample of falling sideways that is predicted as falling backward. Test results revealed three movement patterns that were incorrectly predicted due to striking similarities among them; for instance, the transition from a standing to a squatting position closely resembles the movement from standing to bending over, while falling forward is similar to moving from standing to a prostrate position, and finally, falling sideways closely resembles falling backward. These classification errors indicate that some activities have similar patterns of acceleration and angular velocity changes, making it difficult for the model to distinguish between certain classes. Nevertheless, these results demonstrate that the model is capable of classifying the eight types of activities effectively.

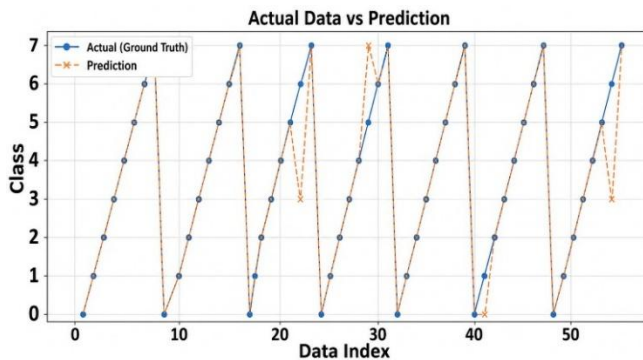


Fig. 7. Actual vs predicted class from all test data.

A comparison is made between the actual labels and the anticipated findings from all test data in order to produce a more comprehensible picture of the categorization results, as shown in Fig. 7. The model prediction results show all patterns nearly identical to the actual data. Most of the predicted points coincide with the actual data, indicating that the model is capable of accurately recognizing the patterns of the eight activity types. Differences are only visible at a few points representing the four

samples that are misclassified, as explained in the confusion matrix results. This indicates that the model performs well in predicting activities based on accelerometer and gyroscope sensor data.

The classification results are then assessed using evaluation metrics, such as accuracy, precision, recall, and specificity, as described in Eq. (6)- (9), to obtain a quantitative estimate of the model performance. Based on the results of the confusion matrix, the model achieved an accuracy of 92.86%, indicating that the majority of the test data is classified correctly. Testing accuracy is higher than training accuracy; this is likely because the test data is smaller, cleaner, or inherently simpler compared to the training dataset. The test data comprises approximately 10% of the total dataset, whereas the training data accounts for 90%. In addition, the model achieved a precision of 94%, a recall of 85%, and a specificity of 97%. These values indicate that the model performs well in recognizing the eight types of activities and is able to accurately distinguish between falls and normal activities.

IV. CONCLUSION

This research successfully developed a comprehensive architecture and algorithm to detect fall and non-fall movement patterns using the backpropagation method within an artificial neural network. The designed system is capable of training on and predicting movement patterns based on data obtained from accelerometer and gyroscope sensors, thereby demonstrating its practical application in real-world scenarios.

Prior to the training process, the raw sensor data are preprocessed by calculating the difference (Δ) between consecutive samples and assigning activity labels to generate more representative input features. The dataset was collected from 69 participants, where data from 62 participants are used for model training and data from the remaining 7 participants are used for model testing. The training process achieved a training accuracy of 86.87% and a validation accuracy of 85.00%, indicating that the proposed model is able to learn the activity patterns effectively while maintaining good generalization performance. The proposed model is evaluated using a confusion matrix on the testing dataset. Experimental results showed that the model correctly classified 52 out of 56 activity samples, achieving an accuracy of 92.86%, precision of 94%, recall of 85%, and specificity of 97%. These findings demonstrate that the proposed backpropagation neural network, combined with the Δ -based preprocessing approach, is capable of accurately classifying eight human movement patterns and has the potential to be applied in inertial sensor-based fall detection system.

ACKNOWLEDGMENT

The Higher Education for Technology and Innovation (HETI) program at the University of Lampung is acknowledged by the author for providing funds for this study.

REFERENCES

- [1] A. Bueno-Cavanilas, F. Padilla-Ruiz, J. J. Jimenez-Moleon, C. A. Peinado-Alonso, and R. Gavez-Vargaz, "Risk factors in falls among the elderly according to extrinsic and intrinsic precipitating causes," *European Journal of Epidemiology*, vol. 16, no. 9, pp. 849-859, Sep. 2000, doi: 10.1023/a:1007636531965.

- [2] A. Bergland and T. B. Wyller, "Risk factors for serious fall-related injury in elderly women living at home," *Injury Prevention*, vol. 10, no. 5, pp. 308-311, Oct. 2004, doi: 10.1136/ip.2003.004721.
- [3] M. T. Aschkenasy and T. C. Rothenhaus, "Trauma and falls in the elderly," *Emergency Medicine Clinics of North America*, vol. 24, no. 2, pp. 413-432, May 2006, doi: 10.1016/j.emc.2006.01.005
- [4] Montero-Odasso M, et al., "World guidelines for falls prevention and management for older adults: a global initiative," *Age Ageing*, vol. 51, no. 9, pp. 1-36, Sep 2022, doi: 10.1093/ageing/afac205.
- [5] F. Khoshnoud and C. W. de Silva, "Recent advances in MEMS sensor technology-mechanical applications," in *IEEE Instrumentation & Measurement Magazine*, vol. 15, no. 2, pp. 14-24, April 2012, doi: 10.1109/MIM.2012.6174574.
- [6] E. Oporto, L.A. Vilcahuaman, "Fall detection system for elderly by MEMS accelerometer and SMS alert," in 2016 VII Latin American Congress on Biomedical Engineering (CLAIB 2016), Bucaramanga, Santander, Colombia, Oct. 2016, pp.114-117, doi: 10.1007/978-981-10-4086-3_29.
- [7] B. Yang, H. Xia, and W. Du, "A detection system for the elderly based on embedded systems," in 2020 35th Youth Academic Annual Conference of Chinese Association of Automation (YAC 2020), Zhanjiang, China, Oct. 2020, pp. 505-508, doi: 10.1109/YAC51587.2020.9337589.
- [8] L. Zheng, J. Zhao, F. Dong, Z. Huang, and D. Zhong, "Fall detection algorithm based on inertial sensor and hierarchical decision," *Sensor*, vol. 23, no. 1, pp. 107/1-107/20, Dec. 2022, doi: 10.3390/s23010107.
- [9] S. S. Jun, H. R. Ramli, A. C. Soh, N. A. Kamsani, R. K. R. Ahmad, S. A. Ahmad, and A. J. Ishak, "Development of fall detection and activity recognition using threshold based method and neural network," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 17, no. 3, pp. 1338-1347, Mar. 2020, doi: 10.11591/ijeecs.v17.i3.pp1338-1347.
- [10] T. Xu, H. Se, and J. Liu, "A fusion fall detection algorithm combining threshold-based method and convolutional neural network," *Microprocessors and Microsystems*, vol. 82, pp. 103828-103837, April 2021, doi: 10.1016/j.micpro.2021.103828.
- [11] S. S. Jeong, N. H. Kim, and Y. S. Yu, "Fall detection system based on simple threshold method and long short-term memory: comparison with hidden markov model and extraction of optimal parameters," *Applied Sciences*, vol. 12, no. 21, pp. 11031-11048, Oct. 2022, doi: 10.3390/app12211031.
- [12] J. Fernandez-Bermejo, J. Martinez-del-Rincon, J. Dorado, X. del Toro, M. J. Santofimia, and J. C. Lopez, "Edge computing transformers for fall detection in older adults," *International Journal of Neural Systems*, vol. 34, no. 5, pp. 2450026/1-2450026/20, May 2024, doi: 10.1142/S0129065724500266.
- [13] F. Wu, H. Zhao, Y. Zhao, and H. Zhong, "Development of a wearable-sensor-based fall detection system," *International Journal of Telemedicine and Applications*, vol. 2015, pp. 576364/1-576364/11, 2015, doi: 10.1155/2015/576364.
- [14] M. N. Farhan, M. G. Ayoub, A. R. H. Al-Jader, "GPS-based fall detection system for old and specially-abled people," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 31, no. 3, pp. 1545-1550, Sep. 2023, doi: 10.11591/ijeecs.v31.i3.pp1545-1550.
- [15] H. Fitriawan, S. Purwiyanti, E. A. Faturrohmah, M. F. Santoso, A. U. Darajat, and T. S. Gunawan, "Development of a low-cost fall detection system for the elderly with accurate detection and real-time alerts," in 2024 IEEE 10th International Conference on Smart Instrumentation, Measurement and Applications (ICSIMA 2024), Bandung, Indonesia, July 2024, pp. 309-314, doi: 10.1109/ICSIMA62563.2024.10675589.
- [16] C.-K. Tseng, S.-J. Huang, and L.-J. Kau, "Wearable fall detection system with real-time localization and notification capabilities," *Sensors*, vol. 25, no. 12, pp. 3632/1-3632/134, Jun. 2025, <https://doi.org/10.3390/s25123632>.
- [17] A. Ibrahim and M. I. Younis, "Simple fall criteria for MEMS sensors: data analysis and sensor concept," *Sensors*, vol. 14, no. 7, pp. 12149-12173, July 2014, doi: 10.3390/s140712149.
- [18] P. V. Er and K. K. Tan, "Non-intrusive fall detection monitoring for the elderly based on fuzzy logic," *Measurement*, vol. 124, pp. 91-102, Aug. 2018, doi: 10.1016/j.measurement.2018.04.009.
- [19] H. Zhang, M. Alrifai, K. Zhou, and H. Hu, "A novel fuzzy logic algorithm for accurate fall detection of smart wristband," *Transactions of the Institute of Measurement and Control*, vol. 42, no. 4, pp. 786-794, Feb. 2020, doi: 10.1177/0142331219881578.
- [20] M. A. Syamlan et al., "Fuzzy logic-based fall detection system for elderly using a single inertial measurement unit," in 2024 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM 2024), Surabaya, Indonesia, Nov. 2024, pp. 1-5, doi: 10.1109/CENIM64038.2024.10882737.
- [21] C. Liu, C. Lee, and P. Lin, "A fall detection system using k-nearest neighbor classifier," *Experts Systems with Applications*, vol. 37, no. 10, pp. 7174-7181, Oct. 2010, doi: 10.1016/j.eswa.2010.04.014.
- [22] P. Bilski, P. Mazurek, J. Wagner, "Application of k nearest neighbors approach to the fall detection of elderly people using depth-based sensors," in The 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS 2015), Warsaw, Poland, Sep. 2015, pp. 722-729, doi: 10.1109/IDAACS.2015.7341400.
- [23] D. Congzhang, Z. Ding, L. Wang, and Y. Jia, "A Fall Detection Method Based on K-Nearest Neighbor Algorithm with MIMO Millimeter-Wave Radar," in 2021 IEEE 6th International Conference on Signal and Image Processing (ICSIP 2021), Nanjing, China, Oct. 2021, pp. 624-628, doi: 10.1109/ICSIP52628.2021.9688752.
- [24] A. Tharwat A, "Classification assessment methods," *Applied Computing and Informatics*, vol. 17, no. 1, pp. 168-192, Jul. 2020, doi: 10.1016/j.aci.2018.08.003.
- [25] J. E. Dobson, "On the confusion matrix," *Configurations*, vol. 32, no. 4, Sep. 2024, pp. 331-350, doi: 10.1353/con.2024.a942087.
- [26] S. Sathyanarayanan and B. R. Tantri, "Confusion matrix-based performance evaluation metrics," *African Journal of Biomedical Research*, vol. 27, no. 4s, pp. 4023-4031, Nov. 2024, doi: 10.53555/AJBR.v27i4S.4345.
- [27] Giovanni Di Franco and Michele Santurro, "Machine learning, artificial neural network and social research," *Quality & Quantity*, vol. 55, no. 3, pp. 1007-1025, Jun. 2021, doi: 10.1007/s11135-020-01037-y.
- [28] Y.-S. Kim, M. K. Kim, N. Fu, J. Liu, J. Wang, J. Srebric, "Investigating the impact of data normalization methods on predicting electricity consumption in a building using different artificial neural network models," *Sustainable Cities and Society*, vol. 118, Jan. 2025, pp. 105570/1-105570/12, doi: 10.1016/j.scs.2024.105570.
- [29] N. Hoang and V. Tran, "A data-driven approach based on deep neural network regression for predicting the compressive strength of steel fiber reinforced concrete," *Journal of Soft Computing in Civil Engineering*, vol. 9, no. 2, pp. 1-31, Spring 2025, doi: 10.22115/scce.2024.430215.1765.
- [30] E. Paluzo-Hidalgo, R. Gonzalez-Díaz, and M. A. Gutiérrez-Naranjo, "Two-hidden-layer feed-forward networks are universal approximators: a constructive approach," *Neural Networks*, vol. 131, pp. 29-36, Nov. 2020, doi: 10.1016/j.neunet.2020.07.021.
- [31] T. Deng, "Effect of the number of hidden layer neurons on the accuracy of the back propagation neural network," in Proceedings of the 2023 4th International Conference on Medical Imaging, Sanitation and Biological Pharmacy (MISBP 2023), Osaka, Japan, Nov. 2023, pp. 462-468, doi: 10.54097/nbra6h45.
- [32] C. Banerjee, T. Mukherjee, and E. Pasilio, "An empirical study on generalizations of the ReLU activation function," in Proceedings of the 2019 ACM Southeast Conference (ACMSE 2019), Association for Computing Machinery, New York, USA, Apr. 2019, pp. 164-167, doi: 10.1145/3299815.3314450.
- [33] G. Luo, X. Wang, W. Zhao, S. Tao, and Z. Tang, "ReLU neural networks and their training," *Mathematics*, vol. 14 no. 1, pp. 39/1-39/12, Jan 2026, doi: 10.3390/math14010039.
- [34] S. Mehra, G. Raut, R. D. Purkayastha, S. K. Vishvakarma, and A. Biasizzo, "An empirical evaluation of enhanced performance softmax function in deep learning," *IEEE Access*, vol. 11, pp. 34912-34924, Jan. 2023, doi: 10.1109/ACCESS.2023.3265327.
- [35] M. Franke and J. Degen, "The softmax function: properties, motivation, and interpretation," *PsyArXiv*, 28 Sept. 2023, doi:10.31234/osf.io/vsw47.