EMITTER International Journal of Engineering Technology
Vol. 10, No. 1, June 2022, pp. 31~46
DOI : 10.24003/emitter.v10i1.639

Towards Improvement of LSTM and SVM Approach for Multiclass Fall Detection System

Herti Miawarni[1], Eko Setijadi[1], Tri Arief Sardjono[1,2], Wijayanti[4], Mauridhi Hery Purnomo[1,3,5]

[1]Dept. of Electrical Engineering, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
[2]Dept. of Biomedical Engineering, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
[3]Dept. of Computer Engineering, Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
[4]Dept. of Arcitecture, Faculty of Engineering, Universitas Diponegoro, Semarang, Indonesia
[5]The Science and Technology Center of Artificial Intelligent for Healthcare and Society (PUI AIHeS), Surabaya, Indonesia
Correspondence Author : hery@ee.its.ac.id

Received January 13, 2022; Revised February 18, 2022; Accepted March 23, 2022

Abstract

Telemonitoring of human physiological data helps detect emergency occurrences for subsequent medical diagnosis in daily living environments. One of the fatal emergencies in falling incidents. The goal of this paper is to detect significant incidents such as falls. The fall detection system is essential for human body movement investigation for medical practitioners, researchers, and healthcare businesses. Accelerometers have been presented as a practical, low-cost, and dependable approach for detecting and predicting outpatient movements in the user. The accurate detection of body movements based on accelerometer data enables the creation of more dependable systems for incorporating long-term development in physiological remarks. This research describes an accelerometer-based platform for detecting users' body movement when they fall. The ADXL345, MMA8451q, and ITG3200 body sensors capture activity data, subsequently classified into 15 fall incident classes based on SisFall dataset. Falling incidents classification is performed using Long Short-Term Memory results in best AUC-ROC value of 97.7% and best calculation time of 6.16 seconds. Meanwhile, Support Vector Machines results in the best AUC-ROC value of 98.5% and best calculation times of 17.05 seconds.

Keywords: Fall Detection System, Sensors-based Monitoring, Accelerometer, Gyroscope, LSTM, SVM.

1. INTRODUCTION

Falls are a serious problem in Indonesia. According to IFLS (Indonesian Family Live Survey) survey data, the falling incidents in the elderly increase every year, namely by 30% in the elderly over 65 years and 50% in the age above 80 years [1]. The elderly with degenerative diseases risk experiencing a higher falling incidents rate, two times the number compared to the elderly who do not have the disease [2]. Several most significant risk factors are namely: muscle weakness and balance disorders, illegal drug consumption, unsafe home environment, visual and hearing impairment, and chronic diseases such as hypertension, stroke, and heart disease [3].

The falling incidents do not only occur in the elderly but can also occur in children and adults. The falling incidents impact the decline in the balance function of falls and can have fatal and non-fatal consequences. Examples of dire consequences caused by falling incidents are permanently paralyzed (disability), psychological effects such as trauma, and fear. In comparison, examples of non-fatal fall incidents are bruising, swelling, and fractures [4].

The purpose of this study is to classify the types of falls on a sensor-based fall incident detection system using a Deep Learning-based algorithm, namely Long Short-Term Memory (LSTM) and tree-based, namely the Support Vector Machine (SVM). The number of types of fall incidents to be classified is 15 classes. The data used in the fall incident detection system is taken from three sensors consisting of two accelerometers sensors, namely the ADXL345 sensor and the MMA8451q sensor, and one gyroscope sensor, the ITG3200 sensor [5]. The LSTM and SVM classification results were evaluated using the True Positive Rate, False Positive Rate, Precision, F-Measure, and Area Under Operating-Receiver Operating Characteristic methods.

The structure of this paper is divided into several parts. Section 1 introduces fall incidents and the importance of building a machine that can automatically recognize fall incidents. Section 2 describes the previous research related to fall incident detection systems using LSTM and SVM and evaluates the classification performance between the two. Furthermore, in Section 3, the sequence of fall incident recognition experiments using Deep Learning (DL) is described. Finally, Section 4 presents the result in performance evaluation from LSTM and SVM and concludes the results of this fall incident detection system.

2. RELATED WORKS

Many types of research on fall incident detection systems have been carried out using various methods, tools, and case studies [6]. Specific for this experiment, we used a dataset that collects data in users' daily lives by monitoring movement and falling incidents, using two accelerometer sensors: ADXL345 sensor and the MMA8451q sensor, and one gyroscope sensor: the ITG3200 sensor. The falling incidents are categorized into 15 classes from these data, and then classification is carried out using LSTM and

SVM. Based on the classification results, LSTM and SVM then underwent a performance evaluation using True Positive Rate, False Positive Rate, Precision, F-Measure, Area Under Operating-Receiver Operating Characteristics.

2.1 Fall Monitoring using Sensors

A fall is regarded as atypical activity, necessitating continuous monitoring of the user's daily life routines. Visual tracking of the user's position is a standard method for obtaining information about their movement in their activity. Overhead camera tracking sends details about user movement trajectories and user activities in designated monitored regions [7], [8].

With an 81 percent success rate for fall detection, Omni-camera images are employed to construct the horizontal arrangement of the user's shadows or silhouettes in the case of falling [9]. With a fall detection success rate of 66.6 percent [10], head tracking is used to follow the user's movement trajectory. The earlier stated solutions for detecting falls in the user based on visual information necessitate collecting equipment and are thus limited to indoor contexts. These circumstances need planned experiments to limit the probability of an accident or fatal harm while performing scripted or unscripted falls.

The usage of sensors that include accelerometers, gyroscopes, and touch sensors is an innovative technique to collect user activity data. There exist research on the use of accelerometers, gyroscopes, and tilt sensors in fall detection [11]. Data acquired from accelerometers are utilized to validate user movement trajectory and time occupancy in defined sections of the experiment and identify abrupt movement related to a fall. This data is often rotation angle or acceleration in the X, Y, and Z axis, using gyroscope sensors. The automatic fall detection is accomplished by the use of established thresholds [12] and the association of current position, movement, and acceleration [11], [13]. This research employed the latter data collection, using two accelerometer sensors, namely the ADXL345 sensor and the MMA8451q sensor, and one gyroscope sensor, namely the ITG3200 sensor. It proves the effective use of accelerometers, resulting in better-than-before classification accuracy if handled with the right Deep Learning models.

2.2 Fall Detection using Deep Learning

Chen et al. created a CNN model with three convolution layers followed by three pooling layers. The convolution kernel of their CNN is customized to the features of the accelerometer data. A total of 31,688 samples were gathered from user behaviors in various experiment regions. Their data is gathered using an Android-based smartphone with an accelerometer sensor. In their study, SVM and Deep Belief Network (DBN) approaches were compared against CNN, with CNN receiving the highest evaluation score of 93.8 percent [14]. Santos et al. explored extensively with CNN models. Their

studies used the Notch dataset and the Smartwatch dataset from prior work, and they constructed a fall detection system with a single convolutional layer. Their best sensitivity score was 91.7 percent [15].

Aicha et al. propose three DL models for fall detection systems: a CNN, an LSTM, and a hybrid of these two models (Convolutional-LSTM or ConvLSTM). An accelerometer sensor was used to collect data on 296 older people ranging in age from 65 to 99 years old. The AUC performance evaluation was used as a performance comparison tool between the Deep Learning models. While both the LSTM and ConvLSTM models outperformed the CNN, the ConvLSTM model has a much shorter learning runtime [16].

Mauldin et al. demonstrated SmartFall, an Android application that serves as an accelerometer and gyroscope sensor for data collecting. Using the Notch dataset, SVM, Naïve Bayes, and Gated Recurrent Unit are used, with the top result reaching 73 percent of TPR [17]. Later on, Zurbuchen et al. used the SVM for their fall detection system, using the public dataset of SisFall [5]. They use wearable sensors to detect users falling to the ground. The SisFall dataset consisted of 15 classes; therefore, the fall detection system is considered a multiclass classification. Although SVM is built for binary classification, ultimately, their SVM reached the TPR score of 87.9 percent and 96.4 in the AUC-ROC score [18].

We chose the single-layer LSTM architecture due to its ability to shorten learning runtime while achieving excellent classification accuracy. As the previous work stated, CNNs and DBNs are outperformed by LSTM. We also made our tuned SVM possible to solve multiclass problems of classification. We also employed the public dataset of activity data consisting of accelerometer and gyroscope data from users falling by design [5]. The LSTM and SVM in this experiment were built from the Java-based machine learning tool. In addition, we added the computation time for both the training stage and testing stage from LSTM and SVM as the performance evaluation.

2.3 Performance Evaluation Efforts

This experiment employed performance evaluation of LSTM and SVM, using the True Positive Rate (TPR), False Positive Rate (FPR), Precision, F-Measure, Area Under Operating-Receiver Operating Characteristic (AUC-ROC) methods, as well as their computation time. We do not employ the accuracy evaluation because the data is already in a balanced class.

TPR is the ratio of the data predicted to be correct with the total actual data, rather than data predicted to be correct [19]. In technical language, the TPR is based on how likely the LSTM and SVM correctly accept the hypothesis, which are the fall categories for the detection system. The TPR is also called Sensitivity or Recall [19]. In contrast, the FPR is based on the probability of rejecting the null hypothesis incorrectly. In statistics, FPR is also called Type I error [19]. Equation 1 and Equation 2 shows basic TPR calculation and basic FPR calculation, respectively.

$$TPR = \frac{True\ Positive}{True\ Positive + False\ Negative} \quad (1)$$

$$FPR = \frac{False\ Positive}{False\ Positive + True\ Negative} \quad (2)$$

Precision is the ratio of the data predicted correctly to the total number of data predicted to be correct [20]. The precision value for each class of falling incidents can be calculated using Equation 3.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \quad (3)$$

The AUC curve is used to measure the performance of the classification algorithm. The higher the AUC value, the better the algorithm's classification ability. A good AUC value has a value close to one. Nevertheless, the classification algorithm performance assessment based on the ROC curve states that a number above 0.5 means that the algorithm has an excellent ability to recognize data in each class [21].

$$F = 2 \times \frac{Precision \times Recall}{TPrecision + Recall} \quad (4)$$

Furthermore, the F-Measure acts as an evaluation to measure the harmonic mean of Precision and Recall [20], as stated in Equation 4. We added the F-Measure methods since the Precision and Recall show an imbalanced weight.

3. ORIGINALITY

We employed LSTM and SVM classification algorithms in this work on data involving falling incidents. Therefore, this work is considered the building of a Fall Detection System (FDS). The SisFall dataset used three sensors to capture humans' motion when falling. As there are many other experiments on FDS, each of them depends on the case and the concern. We specifically build the best Deep Learning models of LSTM and SVM in terms of the accuracy of many performance measures and learning runtime.

Moreover, the SisFall dataset classifies the "fall" into 15 classes, which will hinder the performance of SVM, as it naturally works only for binary classification. However, previous work by Assodiky et al. [22] suggests the adjustment of hyperparameters in respective training models. In this experiment, we manually adjusted the influential hyperparameter settings for LSTM and SVM to fit the case of FDS, which is rarely done.

Machine Learning (ML) improvements aim to reduce computation time and resources. Due to their high performance and outstanding semiotic pattern identification, SVMs have become the most common classification algorithms [23]. In short, our experiment can define contribution as to how excellent Deep Learning models can only measure an excellent dataset. However, the Deep Learning models required hand-picked hyperparameter settings, as the cases vary from one another. SVM is not the best pick for multiclass classification, but we made it a reasonable Deep Learning model for this case by manually adjusting the default hyperparameter of SVM.

Furthermore, a single layer neural network such as LSTM also sought to be in low performance for multiclass classification, in terms of accuracy and runtimes. But we proved it different by improving the hyperparameter setting to fit the SisFall dataset. Overall, we demonstrate the gap for improvement in Deep Learning models in falling incidents cases, especially in testing datasets. We lowered the runtimes for single-layer LSTM and made SVM available for multiclass classification in specific FDS cases.

4. SYSTEM DESIGN

Figure 1 illustrates the experiment setup: firstly, we imported the SisFall dataset into our LSTM and SVM. Then, we processed the data in terms of fall detection. We surely performed several evaluations to ensure the credibility of our LSTM and SVM model output. Lastly, we compared our accuracy results with other FDS works.

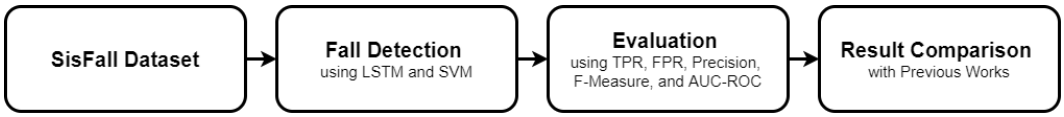

Figure 1. Experiment Setup of Fall Detection System using LSTM and SVM

4.1 Data Collection

This study used a SisFall public dataset about a sensor-based fall incident detection system [5]. Figure 2 illustrates the compilation of SisFall dataset, with two accelerometers' sensors (ADXL345 and MMA8451q) and one gyroscope sensor (ITG3200) [5]. Meanwhile, the details of the dataset are described in Table 1.

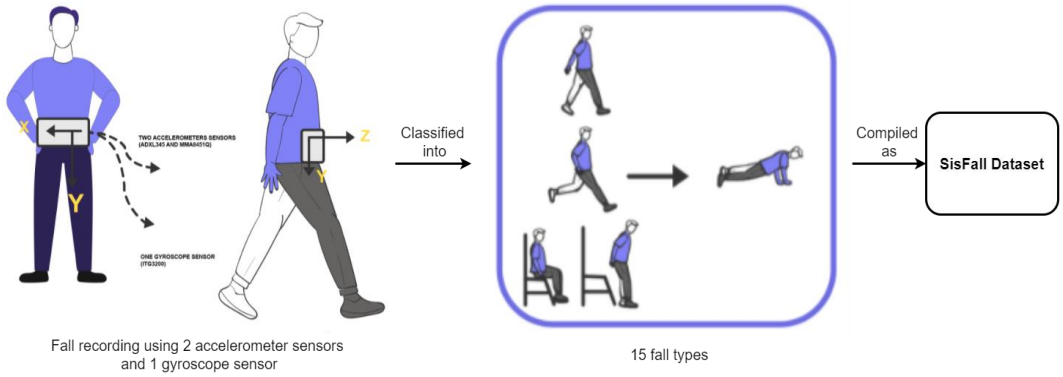

Figure 2. SisFall Dataset which contains of 15 fall types.

Table 1 shows detailed information on the SisFall dataset, consisting of 10 attributes taken from two accelerometer sensors: ADXL345 sensor and MMA8451q sensor with X, Y, and Z axis acceleration, as well as the ITG3200 gyroscope sensor with X, Y, and Z axis rotation data. We have since used the SisFall dataset because it is available to the public, and it is one of the latest falling incident datasets [24]. SisFall is thought to be a ready-for-use dataset, founded in 2017, and its use of accelerometer and gyroscope sensors prove to have an outstanding result for FDS research [25]. There are 15 types of fall incidents classified in SisFall, with 3,000 records for each class/type. The detailed information on the types of falls can be seen in Table 2.

Table 1. Dataset Information

Information	Detail
Dataset	SisFall Dataset [5]
File Type	Comma Separated Values (CSV)
Subjects	23 subjects, ranging from 19 to 30 years old
Amount of Data	45.000 instances
Fall Incident Class(es)	15 classes
Attribute(s)	10 attributes
Data Distribution towards Class(es)	Balanced

Table 2. Fall Types According To SisFall Dataset [5]

Code	Fall Types	Activity	Main Cause
F01	Forward	Walking	Slipped
F02	Backward	Walking	Slipped
F03	Lateral	Walking	Slipped
F04	Forward	Walking	Tripped
F05	Forward	Jogging	Tripped
F06	Vertical	Walking	Fainted
F07	Fall-breaking	Walking	Fainted
F08	Forward	Trying to get up	-

Code	Fall Types	Activity	Main Cause
F09	Lateral	Trying to get up	-
F10	Forward	Trying to sit	Activity
F11	Backward	Trying to sit	Activity
F12	Lateral	Trying to sit	Activity
F13	Forward	Sitting down	Fainted/Fell Asleep
F14	Backward	Sitting down	Fainted/Fell Asleep
F15	Lateral	Sitting down	Fainted/Fell Asleep

4.2 Data Pre-Processing

To fully maximize the use of the Deep Learning models, the data firstly need to be "cleaned". We used a community-recommended data pre-processing: the Replace Missing Value package [22]. Missing data treatment is critical for avoiding skewed outcomes [26]. In general, missing data in the SisFall dataset lowers the power of the LSTM and SVM models, resulting in incorrect fall detection. Even missing value columns and rows frequently play a significant influence in the LSTM and SVM output. We certainly don't want to miss any records in the columns and rows essential to the LSTM and SVM learning processes. Leaving these factors out could result in redundancy in our results. Consequently, missing value treatment is critical for developing the best-fit LSTM and SVM models. It is a part in the direction of adjusting data and learning models, which is the focus of this experiment: to build a learning model for FDS purposes.

Both LSTM and SVM data are prepared by first normalizing data using respective built-in functions. For LSTM, the data normalization function is located before data input in the layer of LSTM itself. In contrast, for SVM, the data normalization function is located before data input and in the kernel. This kernel is named as Normalized Polynomial Kernel [27]. Furthermore, the data is split into two, with the training stage getting 80% of the total data, and the testing stage getting 20% of the total data.

4.3 Fall Detection using LSTM and SVM

For a successful detection procedure, finding and using the appropriate training settings is a critical first step. As a result, numerous hyperparameter adjustments should be made. These hyperparameter settings are activation functions, learning rate, early stopping condition, size of the batch, dense constant, and loss function for LSTM. For SVM, these hyperparameter adjustment is made for attribute normalization type, batch size, calibrator and complexity, gradient descent, epsilon constant and kernel type, iteration, ridge constant, and tolerance parameter. Table 3 shows the one-layer LSTM architecture built for our fall detection experiment, while Table 4 shows the LSTM hyperparameter setting used in this experiment. Moreover, Table 5 shows the hyperparameter setting for the SVM setup.

Table 3. LSTM Architecture Model

Layer	**Input/Output**	**Total Parameters**	**Parameters Shape**
LSTM	9/15	1.500	W: {9, 60} RW: {15, 60} B: {1, 60}
Dense	15/15	240	W: {15, 15} B: {1, 15}
Output	15/15	240	W: {15, 15} B: {1, 15}
Total Parameters	**1.980**		
Trainable Parameters	**1.980**		
Non-Trainable Parameters	**0**		

Table 4 and Table 5 outline each hyperparameter's details and its associated function. For the efficiency of the training phase, the respective model's hyperparameters must be adjusted. Furthermore, a backpropagation update is used to train the LSTM model. The goal of this update is to reduce the loss function to zero in each iteration.

After that, we'll look at the optimal learning rate for building the LSTM model. Specifically, this is performed by varying the step size used to update the model's weights with respect to the loss function. Choosing the most appropriate learning rate can be difficult, and this is a problem for many researchers. An inefficient convergence rate may result in a progressive decline of the loss function, or an inefficient convergence rate may inhibit convergence. It could cause a variety of loss functions. These hyperparameters are considered the critical parameters that must be adjusted throughout the LSTM's training process. On the other hand, our adjusted SVM is not prone to overfitting and adequately handles high dimensional data. SVM also benefits from being memory-efficient, making it ideal for accelerometer and gyroscope sensors.

Table 4. LSTM Hyperparameter Setup

Hyperparameter	**Value/Function**
Activation Function	ReLU
Attribute Normalization Type	Normalize training data
Batch Size	100
Dense Activation Function	Softmax
Early Stopping	False
Enable Intermediate Evaluation	True
Epoch	50
Epoch Listener	True (5)
Gate Activation Function	Sigmoid
Loss Function	MCXENT
Output Activation Function	Softmax

Table 5. SVM Hyperparameter Setup

Hyperparameter	**Value/Function**
Attribute Normalization Type	Normalize training data
Batch	100
Calibrator	Logistic Regression
Complexity Parameter	1.0
Conjugate Gradient Descent	True
Epsilon	1.0E-12
Kernel	Normalized Polynomial
Maximum Iteration (Calibrator)	-1
Ridge (Calibrator)	1.0E-8
Tolerance Parameter	0.001

5. EXPERIMENT AND ANALYSIS

This section presents our experiment results and its explanation in the form of discussion.

5.1 Fall Detection using LSTM and SVM

Table 6 shows the various validation based on fall detection performance using LSTM, while Table 7 shows the validation based on SVM performance. Table 6 and Table 7 have remarks for TPR, FPR, Precision, F-Measure, and AUC-ROC changed to Evaluation 1st to Evaluation 5th. Meanwhile, Figure 3 illustrates the chart of the training stage and testing stage computation time of both LSTM and SVM.

Table 6. Validation Based on LSTM Performance

Class	**Evaluation**				
	1st	**2nd**	**3rd**	**4th**	**5th**
F01	0.810	0.015	0.795	0.802	0.955
F02	0.161	0.040	0.226	0.188	0.823
F03	0.467	0.013	0.717	0.565	0.922
F04	0.713	0.048	0.498	0.586	0.924
F05	0.695	0.029	0.638	0.665	0.919
F06	0.474	0.055	0.382	0.423	0.907
F07	0.618	0.019	0.706	0.659	0.926
F08	0.833	0.007	0.886	0.859	0.960
F09	0.799	**0.005**	0.924	0.857	0.953
F10	**0.925**	0.030	0.695	0.794	**0.977**
F11	0.640	0.031	0.586	0.612	0.954
F12	0.572	0.009	0.830	0.678	0.920
F13	0.835	0.016	0.783	0.808	0.971
F14	0.744	0.023	0.710	0.727	0.968
F15	0.859	0.008	0.889	0.874	0.967

Table 7. Validation Based on SVM Performance

Class	Evaluation				
	1st	2nd	3rd	4th	5th
F01	0.523	0.016	0.696	0.597	0.849
F02	0.365	0.022	0.547	0.438	0.853
F03	0.462	**0.008**	0.805	0.587	0.853
F04	0.144	0.033	0.226	0.176	0.835
F05	0.643	0.054	0.463	0.538	0.878
F06	0.509	0.012	0.748	0.606	0.884
F07	0.238	0.085	0.168	0.197	0.844
F08	0.007	0.041	0.011	0.009	0.861
F09	0.855	0.110	0.367	0.514	0.920
F10	0.429	0.066	0.322	0.368	0.861
F11	0.572	0.028	0.591	0.581	0.893
F12	0.036	0.018	0.125	0.056	0.929
F13	0.569	0.026	0.602	0.585	0.919
F14	**0.932**	0.030	0.695	0.796	**0.985**
F15	0.592	0.027	0.619	0.605	0.958

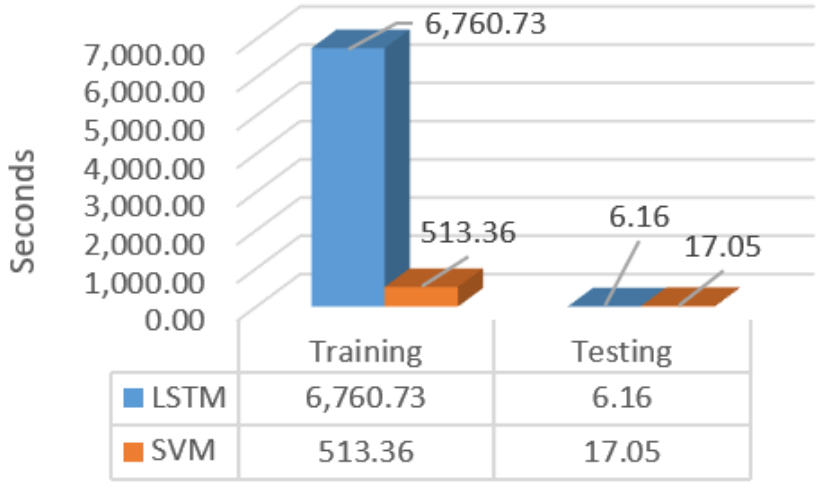

Figure 3. Computation Time Comparison of LSTM and SVM (in seconds)

Over the years, researchers have taken a variety of learning models for fall detection systems. These previous works resulted in a certain level of accuracy. Then, we compared our LSTM and SVM results with these previous works. Our LSTM and SVM prove to have excellent accuracy of 98.5% for LSTM and 97.7% for SVM. The comparison of our work and other FDS works is shown in Table 8.

Table 8. Comparison of Our Proposed Model and other FDS works

Previous Work	Year	Learning Model	Accuracy
Chen et al. [14]	2015	SVM-DBN	93.8%
Pierloni et al. [28]	2015	SVM	95.18%
Mauldin et al. [17]	2018	SVM	73%
Wisesa et al. [29]	2019	LSTM	86.63%
Santos et al. [15]	2019	SVM-CNN	91.7%
Zurbuchen et al. [18]	2020	SVM	96.4%
Our Proposed Models	**2021**	**SVM**	**98.5%**
		LSTM	**97.7%**

5.2 Discussion

Based on the result of LSTM and SVM performance validation, it is clear that SVM can process the SisFall dataset into a TPR score of 0.932 or 93.2% in percentage. With only 0.013 or 0.13% difference, LSTM could catch up SVM's TPR score by 0.925 or 92.5% in percentage. However, LSTM has the best FPR score with 0.005, while SVM is 0.008. Furthermore, both DL models showed no struggle in this fall detection system, as the LSTM score of AUC-ROC is 0.977 or 97.7%, while the SVM score of AUC-ROC is 0.985 or 98.5%. This finding means that both LSTM and SVM are able to tell the difference between each of the fall classes. AUC-ROC rules suggest that any result above 50% means that the model has an outstanding capability to learn between classes [21]. When confronted with extended sequences, LSTMs tend to prioritize neighboring information. It is quite susceptible to gradient dispersion. As a result, if the data sequence is lengthy, the data usage rate will be inadequate.

Another performance evaluation, such as computation time, is also done for both LSTM and SVM. According to our experiment, SVM achieved 513.36 seconds (roughly 8.5 minutes) for the training stage and 17.05 seconds for the testing stage. LSTM has a slower training time by 6,760.73 seconds or roughly 1.87 hours, but blazing fast testing time by only 6.16 seconds. In Figure 3, LSTM shows a slower training time than SVM. This is natural for LSTM. Slower computation time does not always mean that the model will perform better and be more accurate [30]. The same goes for faster computation time. However, we found that a slower training time is aligned with a faster testing time. The more time it takes for a model in the training stage, the faster it will become in the testing stage.

6. CONCLUSION

This work presents the fall detection system using LSTM and SVM as the Deep Learning models. LSTM is picked as previous work stated that LSTM

outperforms most vanilla Deep Learning models, including CNN. Meanwhile, SVM was chosen to face a classification problem that is not natural for SVM, namely the multiclass problem in standard classification. As for the dataset, we used the SisFall dataset, which comprises 3,000 rows of fall information based on three sensors, two of which are accelerometer sensors (ADXL345 and MMA8451q) and one gyroscope sensor (ITG3200). All sensors yield information on the user's X, Y, and Z axis positions. Moreover, this dataset also represents a multiclass dataset; it has 15 classes of falling incidents with different fall types, activities, and main causes.

SVM (in its general purpose) cannot be implemented in multiclass classification. Single-layer LSTM, on the other hand, also cannot be beneficial in handling more than 15 classes of sensor-based data of fall detection. However, in this paper, we were able to achieve the best result with a bit of experiment of modifying hyperparameter value. Based on our experiment, LSTM can yield a sensitivity score of 92.5%, while SVM can yield a higher value of 93.2%. Using FPR, our LSTM has reached 0.005, and SVM has reached 0.008. Both DL models also have relatively fast runtimes in the testing stage, with LSTM peaked in 6.16 seconds and SVM peaked in 17.05 seconds. These statistical accuracies are presented to evaluate how the LSTM and SVM are able to handle the multiclass SisFall dataset. These results also show the ability of SVM to handle multiclass classification for a fall detection system under our custom-picked hyperparameter setting. However, for SVM to perform much better in multiclass problems, one needs to revamp the entire mathematical aspect of SVM, which is essentially the same as not using SVM and creating a new kind of tree-based Deep Learning model. The result also indicates that the single-layer neural network of LSTM can handle such a large multiclass dataset under fair learning runtime.

We believe in a better result by tinkering further to the respective DL model's hyperparameter settings, including carefully selecting the kernels. Changing the mathematical aspect of SVM may make it even more robust in handling much more classes and data samples. Adding other layers into LSTM architecture and combining them with different layers is also recommended for future work purposes.

Acknowledgements

We thank the Kementerian Riset dan Teknologi (Indonesian Ministry of Research and Technology) and the Badan Riset dan Inovasi Nasional (Indonesian National Research and Innovation Agency) for providing the research grant with master contract number 3/E1/KP.PTNBH/2021, March 8, 2021 and researcher contract number 881/PKS/ITS/2021, march 10, 2021.

REFERENCES

[1] J. Strauss, F. Witoelar, and B. Sikoki, **The fifth wave of the Indonesia family life survey: overview and field report**, vol. 1. RAND Santa

Monica, CA, 2016.
[2] S. Nugraha, I. Hapsari, S. Pengpid, K. Peltzer, and others, **Multimorbidity Increases the Risk of Falling among Indonesian Elderly Living in Community Dwelling and Elderly Home: A Cross Sectional Study**, *Indian J. Public Heal. Res. \& Dev.*, vol. 10, no. 11, 2019.
[3] S. Pengpid and K. Peltzer, **Prevalence and risk factors associated with injurious falls among community-dwelling older adults in Indonesia**, *Curr. Gerontol. Geriatr. Res.*, vol. 2018, 2018.
[4] N. Zurbuchen, A. Wilde, and P. Bruegger, **A Machine Learning Multi-Class Approach for Fall Detection Systems Based on Wearable Sensors with a Study on Sampling Rates Selection**, *Sensors*, vol. 21, no. 3, p. 938, Jan. 2021, doi: 10.3390/s21030938.
[5] A. Sucerquia, J. D. López, and J. F. Vargas-Bonilla, **SisFall: A fall and movement dataset**, *Sensors*, vol. 17, no. 1, p. 198, 2017.
[6] H. Miawarni *et al.*, **Fall Detection System for Elderly based on 2D LiDAR: A Preliminary Study of Fall Incident and Activities of Daily Living (ADL) Detection**, in *2020 International Conference on Computer Engineering, Network, and Intelligent Multimedia (CENIM)*, Nov. 2020, pp. 1–5, doi: 10.1109/CENIM51130.2020.9298000.
[7] H. Nait-Charif and S. J. McKenna, **Activity summarisation and fall detection in a supportive home environment**, in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, 2004, vol. 4, pp. 323–326.
[8] B. Jansen and R. Deklerck, **Context aware inactivity recognition for visual fall detection**, in *2006 Pervasive Health Conference and Workshops*, 2006, pp. 1–4.
[9] S.-G. Miaou, P.-H. Sung, and C.-Y. Huang, **A customized human fall detection system using omni-camera images and personal information**, in *1st Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare, 2006. D2H2.*, 2006, pp. 39–42.
[10] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, **Monocular 3D head tracking to detect falls of elderly people**, in *2006 international conference of the IEEE engineering in medicine and biology society*, 2006, pp. 6384–6387.
[11] F. R. Allen, E. Ambikairajah, N. H. Lovell, and B. G. Celler, **An adapted gaussian mixture model approach to accelerometry-based movement classification using time-domain features**, in *2006 International Conference of the IEEE Engineering in Medicine and Biology Society*, 2006, pp. 3600–3603.
[12] M. Prado, J. Reina-Tosina, and L. Roa, **Distributed intelligent architecture for falling detection and physical activity analysis in the elderly**, in *Proceedings of the Second Joint 24th Annual Conference and the Annual Fall Meeting of the Biomedical Engineering Society][Engineering in Medicine and Biology*, 2002, vol. 3, pp. 1910–1911.

[13] S. Luo and Q. Hu, **A dynamic motion pattern analysis approach to fall detection**, in *IEEE International Workshop on Biomedical Circuits and Systems, 2004.*, 2004, pp. 1–5.
[14] Y. Chen and Y. Xue, **A deep learning approach to human activity recognition based on single accelerometer**, in *2015 ieee international conference on systems, man, and cybernetics*, 2015, pp. 1488–1492.
[15] G. Santos, P. Endo, K. Monteiro, E. Rocha, I. Silva, and T. Lynn, **Accelerometer-Based Human Fall Detection Using Convolutional Neural Networks**, *Sensors*, vol. 19, no. 7, p. 1644, Apr. 2019, doi: 10.3390/s19071644.
[16] A. Nait Aicha, G. Englebienne, K. S. Van Schooten, M. Pijnappels, and B. Kröse, **Deep learning to predict falls in older adults based on daily-life trunk accelerometry**, *Sensors*, vol. 18, no. 5, p. 1654, 2018.
[17] T. R. Mauldin, M. E. Canby, V. Metsis, A. H. H. Ngu, and C. C. Rivera, **SmartFall: A smartwatch-based fall detection system using deep learning**, *Sensors*, vol. 18, no. 10, p. 3363, 2018.
[18] N. Zurbuchen, P. Bruegger, and A. Wilde, **A Comparison of Machine Learning Algorithms for Fall Detection using Wearable Sensors**, in *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, Feb. 2020, pp. 427–431, doi: 10.1109/ICAIIC48513.2020.9065205.
[19] D. van Ravenzwaaij and J. P. A. Ioannidis, **True and false positive rates for different criteria of evaluating statistical evidence from clinical trials**, *BMC Med. Res. Methodol.*, vol. 19, no. 1, pp. 1–10, 2019.
[20] D. M. W. Powers, **Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation**, *arXiv Prepr. arXiv2010.16061*, 2020.
[21] S. Narkhede, **Understanding AUC - ROC Curve**, *Towards Data Science*, 2018. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5.
[22] H. Assodiky, I. Syarif, and T. Badriyah, **Arrhythmia Classification Using Long Short-Term Memory with Adaptive Learning Rate**, *Emit. Int. J. Eng. Technol.*, vol. 6, no. 1, pp. 75–91, Jul. 2018, doi: 10.24003/emitter.v6i1.265.
[23] D. P. Adi, L. Junaedi, Frismanda, A. B. Gumelar, and A. A. Kristanto, **Exploring the Time-efficient Evolutionary-based Feature Selection Algorithms for Speech Data under Stressful Work Condition**, *Emit. Int. J. Eng. Technol.*, vol. 9, no. 1, pp. 60–74, Feb. 2021, doi: 10.24003/emitter.v9i1.571.
[24] E. Casilari, J.-A. Santoyo-Ramón, and J.-M. Cano-García, **Analysis of Public Datasets for Wearable Fall Detection Systems**, *Sensors*, vol. 17, no. 7, p. 1513, Jun. 2017, doi: 10.3390/s17071513.
[25] C. Krupitzer, T. Sztyler, J. Edinger, M. Breitbach, H. Stuckenschmidt, and C. Becker, **Beyond Position-Awareness—Extending a Self-Adaptive**

Fall Detection System, *Pervasive Mob. Comput.*, vol. 58, p. 101026, Aug. 2019, doi: 10.1016/j.pmcj.2019.05.007.

[26] M. L. Yadav and B. Roychoudhury, **Handling Missing Values: A Study of Popular Imputation Packages in R**, *Knowledge-Based Syst.*, vol. 160, pp. 104–118, Nov. 2018, doi: 10.1016/j.knosys.2018.06.012.

[27] P. H. Prastyo, A. S. Sumi, A. W. Dian, and A. E. Permanasari, **Tweets Responding to the Indonesian Government's Handling of COVID-19: Sentiment Analysis Using SVM with Normalized Poly Kernel**, *J. Inf. Syst. Eng. Bus. Intell.*, vol. 6, no. 2, pp. 112–122, 2020.

[28] P. Pierleoni, A. Belli, L. Palma, M. Pellegrini, L. Pernini, and S. Valenti, **A High Reliability Wearable Device for Elderly Fall Detection**, *IEEE Sens. J.*, vol. 15, no. 8, pp. 4544–4553, 2015, doi: 10.1109/JSEN.2015.2423562.

[29] I. Wayan Wiprayoga Wisesa and G. Mahardika, **Fall detection algorithm based on accelerometer and gyroscope sensor data using Recurrent Neural Networks**, *IOP Conf. Ser. Earth Environ. Sci.*, vol. 258, no. 1, 2019, doi: 10.1088/1755-1315/258/1/012035.

[30] W. Yu, X. Li, and J. Gonzalez, **Fast Training of Deep LSTM Networks**, in *International Symposium on Neural Networks*, 2019, pp. 3–10.