International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. #, No.# (Mon-20..)

Enhancing Smartphone Motion Sensing with Embedded Deep Learning

Panduranga Vital Terlapu[1,*], Jayaram D[2]

[1,*] *Department of Computer Science and Engineering, Aditya Institute of Technology and Management, Tekkali, Srikakulam, Andhra Pradesh-532 201, India*
[2]*Department of Information Technology, Chaitanya Bharathi Institute of Technology, Hyderabad , Telangana , India.*

E-mail address: vital2927@gmail.com

Received ## Mon. 20##, Revised ## Mon. 20##, Accepted ## Mon. 20##, Published ## Mon. 20##

Abstract: Embedded systems and smartphones are vital in real-time applications, inspiring our interaction with technology. Smartphones possess various sensors like accelerometers, gyroscopes, and magnetometers. Deep Learning (DL) models enhance the capabilities of sensors, enabling them to perform real-time analysis and decision-making with accuracy and speed. This study demonstrates an intelligent system that detects smartphone movements using deep learning (DL) techniques such as convolutional neural networks (CNN) and stacked autoencoders(SAEs). The dataset has six smartphone movements, with 921 samples split into 695 for training and 226 for testing. The best training performance was achieved by Auto-Encoder 1 and Auto-Encoder 2. The SAEs had high classification accuracy (CA) and AUC values of 0.996 and 1.0, respectively. Similarly, CNN performed well with CA and AUC values of 0.991 and 0.998. These results show that CNN and SAEs are effective in identifying smartphone movements. The findings help improve smartphone apps and understand how well they can identify movement. The study indicates that CNN and SAE are influential in accurately identifying smartphone movements. Future research can improve motion detection by integrating more sensor data and advanced models. Using advanced deep learning architectures like RNNs or transformers can enhance the understanding and accuracy of predicting smartphone movements.

Keywords: Auto-Encoder, CNN, Embedded Systems, Machine Learning, Smart Phone, Sensor Data

1. Introduction

Smartphones are ubiquitous and have many features [1]. They have accelerometers, gyroscopes, and magnetometers that track movement and orientation. By using DL models, we can create intelligent apps to analyze this sensor data (SD) in real time and make decisions. The "Intelligent Smart Phone Movement Identification Embedded System" is a system that uses smartphone sensors to recognize and classify smartphone movements accurately. It uses advanced DL models to analyze the SD and detect user movements. The system uses the smartphone's magnetometer, gyroscope, and accelerometer to record precise motion data in real time. It relies on real-time embedded systems to process the SD quickly and efficiently. The system has sensor interfaces, data acquisition modules, processing units, and decision-making algorithms. These components work together to read the SD and interpret the movements. The sensors continuously record the phone's orientation, acceleration, and angular velocity, providing a lot of data to understand how the user moves. Once the movements are identified, the embedded system can act or give feedback to the user or other apps on the smartphone [2][3]. DL models are used in this system to process and understand SD. These models learn and recognize different movement patterns from a large amount of labelled data. The system can identify smartphone gestures such as clock and anti-clock cycles, up-down, left-right, wave or snake cycles, and idle mode. It accurately distinguishes these movements and provides real-time feedback on smartphone movements. This system has many benefits, including tracking physical activities without wearable devices [4]. It eliminates the need for bulky accessories and improves user convenience.

The medical and fitness industries use these systems that monitor physical activity patterns and assist people in becoming more fit. It combines DL with smartphones to accurately identify and analyze movement. The research aims to address challenges in smartphone movement identification and proposes an intelligent solution using embedded DL models. The paper will review existing literature, discuss advancements and limitations, and present the proposed model and materials used for experimentation. The paper's remaining sections are described as

- The **literature review** thoroughly examines previous research and publications on smartphone movement identification and embedded DL models. Researchers' methods, algorithms, and techniques will be discussed in this. The review identifies literature gaps, challenges, and opportunities to form our model.
- **Proposed Model and Materials**: We present our intelligent smartphone movement identification model using embedded DL. Our model's training method is architecture and network design. The dataset, pre-processing methods, and hardware and software requirements for our experiments will also be discussed.
- The experimental results of our model are shown in the **results and discussion** sections. We evaluated our approaches using performance metrics like F1 score, accuracy, precision, and recall. By comparing our findings to methods reviewed in the literature, we can identify the strengths and weaknesses of our model. We will provide a detailed explanation of the reasons behind the results.
- **Conclusion:** We summarize our intelligent smartphone movement identification research using embedded DL models. We emphasize the importance of our approach and its potential impact on field applications. Our findings, limitations, and future research will be discussed.

2. LITERATURE REVIEW

Smartphone technology has improved and now allows for new uses in recognizing human activities and healthcare applications. Researchers Huang et al. [5] have suggested using a cell phone dongle for blood lipid testing. The dongle has shown promising results, with high correlation coefficients (0.903) and low variation (4.575%). It indicates that the dongle is accurate and reliable, making it a valuable tool for cholesterol testing in the future. Johnson et al. [6] have developed an algorithm that can automatically detect whether a smartphone is used by the driver or passenger in a moving vehicle. This technology can help prevent distracted driving, especially for iPhones and other smartphones that currently cannot differentiate between driver and passenger usage. Masud et al. [7] have introduced a smartphone-based approach to assess depression levels by monitoring daily activities. This method is cost-effective and non-intrusive and has achieved high accuracy (87.2 %) in identifying severe depression cases. It offers a promising way to assess and monitor depression, providing timely intervention and support for individuals with mental health disorders. Qi et al. [8] presented a smartphone-based human activity recognition (HAR) and automatic labelling framework. This framework uses signals from Microsoft Kinect cameras and smartphones to label everyday tasks accurately. Compared to other models, their methodology and algorithm perform better in accuracy assessment.

Researchers studied and discussed a method called DL-based sensor-based activity recognition. They also proposed a FL-PMI approach to improve the accuracy and efficiency of intelligent healthcare systems. Wang et al. [9] conducted a survey on DL for sensor-based activity recognition. They researched recent advancements, challenges, and potential solutions in DL-based activity recognition. The survey discussed sensor modality, deep models, and applications in detail. It also provided insights into the factors contributing to improved performance in DL models. The survey listed common public datasets used for research in activity recognition and discussed unique problems and solutions in DL-based activity recognition. Arikumar et al. [10] proposed a method called FL-PMI that combines federated learning, deep reinforcement learning, and bidirectional long short-term memory. Their method achieved high accuracy and addressed computational costs, memory usage, and data transmission challenges in intelligent healthcare systems. It improved efficiency, reduced resource requirements, and decreased data transmission by 36.73%. The FL-PMI outperformed other systems in accuracy, precision, F1-score, and recall, demonstrating its effectiveness in sensor-based activity recognition. Liu et al. [11] presented a security mechanism using channel state information to detect rogue Wi-Fi devices. Their mechanism achieved a high accuracy of 96% in detecting rogue connections and had a low false alarm rate. It was also eight times faster than existing solutions regarding detection speed. This research enhances security and mitigates vulnerabilities in Wi-Fi networks, WLAN, and IoT environments. The study [12] addressed vulnerabilities in biometric-based authentication methods by introducing Lip Pass, a lip-reading-based authentication system. Their DL approach, combined with Doppler profiles and smartphone acoustic sensing, achieved high accuracy rates in user identification and spoofed detection. Lip-reading-based authentication can enhance privacy protection on mobile devices and is resilient to ambient environmental factors.

Recent research has focused on how to use DL techniques, like CNNs, to make systems more accurate and efficient. Table I summarizes research on smartphone sensor data (SSD) for HAR using different ML and DL models. Studies focus on dataset size, sensor types, model architecture, and performance metrics.

TABLE I SOME OF THE RESEARCH WORKS ON SMART-PHONE SENSOR DATA

Ref. No.	Author &Year	Aim and Description	Results and Models Analysis
[13]	Jiang et al.	The study utilizes a smartphone-based HAR system, utilizing CNN, to investigate the correlation between health issues and six physical activities.	The CNN achieved a recognition accuracy of 97.5% for a UCI HAR dataset.
[14]	Zhou et al.	The study developed a smartphone-based HAR using CNN, synthesizing a 6 GB sensor dataset from various smartphones' accelerometers, magnetometers, gyroscopes, and barometers.	CNN achieved 97.5% accuracy, while J48 achieved 95.5% accuracy.
[15]	Ravi et al.	The proposed DL approach utilizes SD analytics on wearable or mobile devices for efficient on-node processing, utilizing CNN models on diverse datasets.	Active-Miles achieved a 95.7% accuracy rate, WISDM v1.1-98.6%, WISDM v2.0-92.7%, Skoda-95.3%, and DaphnetFoG-95.8% for each data set.
[16]	Zebin et al.	The research evaluated the DL system's memory and execution time HAR system on mid-range smartphone hardware, focusing on designing a CNN for HAR tasks.	The proposed CNN model demonstrated a remarkable accuracy(96.4%) in a five-class static and dynamic activity recognition scenario.
[17]	Qi et al.	The authors developed a novel FR-DCNN model for HAR, utilizing smartphones, ISP algorithms, and an SS module to improve the effectiveness and extend the entropy of IMU sensors' raw data.	The FR-DCNN model demonstrated a remarkable prediction time of 0.0029 seconds with a 95.27% accuracy for HAR.
[18]	Xia et al.	The authors developed a DNN that integrates CLs and LSTM for automatic feature extraction and classification of activity data, resulting in significant enhancements.	The model demonstrated high accuracy on three public datasets, with 95.78% accuracy on UCI-HAR, 95.85% accuracy on WISDM, and 92.63% accuracy on OPPORTUNITY.
[19]	Ye et al.	The authors explored the performance limits of combining two-stream and recurrent neural networks, highlighting spatial structure and appropriate fusion methods, and proposing ConvLSTM networks with two-stream ConvNet.	The proposed method achieves accuracy rate (69.4%) on HMDB51 and 93.9% on UCF101 datasets, demonstrating its effectiveness in activity recognition tasks.

Ponciano et al. [20] have developed a method using smartphones and sensors to measure and analyze the timed-up and going test parameters. This method makes it easier to collect and process data, which can be used to group test results and identify patterns related to balance changes, neurological disorders, and other conditions. The proposed architecture shows promise for comprehensive analysis and assessment in physiotherapy. The study [21] proposed a deep neural network architecture for HAR that encodes SD as images and uses computer vision (CV) techniques. This method performs better than other methods in terms of F1-value and accuracy rate by using fusion residual networks and different layers of deep residual networks. Testing on various datasets supports the idea that this method can help make data-driven decisions for HAR. The comparative analysis of fusion layers using a specific dataset shows that the Conv3 fusion layer performed the best with high accuracy. Dasgupta et al. [22] studied how economic and healthcare factors in US counties were related and how social distancing measures affected people's movement. They used data from mobile devices to analyze this. The study found that places with stay-at-home orders significantly decreased movement than places without orders. This showed that some people had more privilege to stay home than others. The study shows that social distancing is essential in addressing inequalities during health crises like COVID-19.

Auto-encoders are models that can learn from data without supervision. They can be used for tasks like reducing dimensions, detecting anomalies, and extracting features. They are good at finding patterns in data. They can be used to detect human activities using SSD. A study by Alo et al. [23] showed that their deep-stacked auto-encoder algorithm, along with features that don't change with orientation, can accurately identify complex human activities with a high level of accuracy. This is better than traditional ML methods and deep belief networks. This approach can be helpful for health monitoring, fall detection, and emotion detection. Another study by Garcia et al. [24] introduced a multi-class algorithm that uses an ensemble of auto-encoders for HAR. The results showed that this approach is more effective than other techniques. The group of autonomous auto-encoders had a higher accuracy in comparison to other models like EkVN. The autonomous auto-encoders also outperformed EkVN regarding user accuracy, showing better performance in most users. However, different models like JiangYin and Haetal slightly outperformed the autonomous auto-encoders and EkVN regarding average accuracy. Wu et al. [25] emphasized the importance of road surface maintenance and proposed an automatic pothole detection system using smartphone vibration sensors.

Researchers Hasanuzzaman et al. [26] have studied DL -based HAR systems in wearable devices and smartphones. They found that these systems have unique features, advantages, and limitations. One study [26] focused on the transpiration influences on the flow of a vertical, shallow body's boundaries in natural convection. They used numerical analysis to understand the impact of parameters on fluid dynamics and heat transfer. Another study [27] reviewed DNN methodologies for automatic feature extraction in HAR. They discussed the implementation, design, advantages, and disadvantages of these methodologies, as well as their performance evaluation. The study also highlighted challenges and discussed various DL functionalities used in sensor-based recognition systems for wearables and smartphones, providing insights into their uniqueness, advantages, and limitations.

In a study by Rana et al.[28], a system was developed to monitor and assess the roughness of road surfaces using vehicle dynamics and smartphones. This system aims to improve road maintenance and driving conditions. Kołakowska et al. [29] conducted a comprehensive review of emotion recognition methods using SSD data. Their analysis provides insights into advancements, challenges, and potential applications in the field of emotion detection in mobile computing. Nooruddin et al. [30] reviewed fall detection systems to help keep older adults and those prone to falls safe. They discussed various sensors, algorithms, and techniques, along with the challenges and future directions in the field. Straczkiewicz et al. [31] systematically reviewed smartphone-based HAR for healthcare purposes. They analyzed the existing literature and highlighted these methods' strengths, limitations, and potential applications in monitoring and analyzing HAR.

3. Methodology and Materials

The suggested model intends to produce an intelligent embedded system that accurately classifies six smartphone movements using DL techniques. It uses CNNs and SAEs models to process SD, enabling smartphones to adapt to user movements, improving user experiences and mobile app interaction.

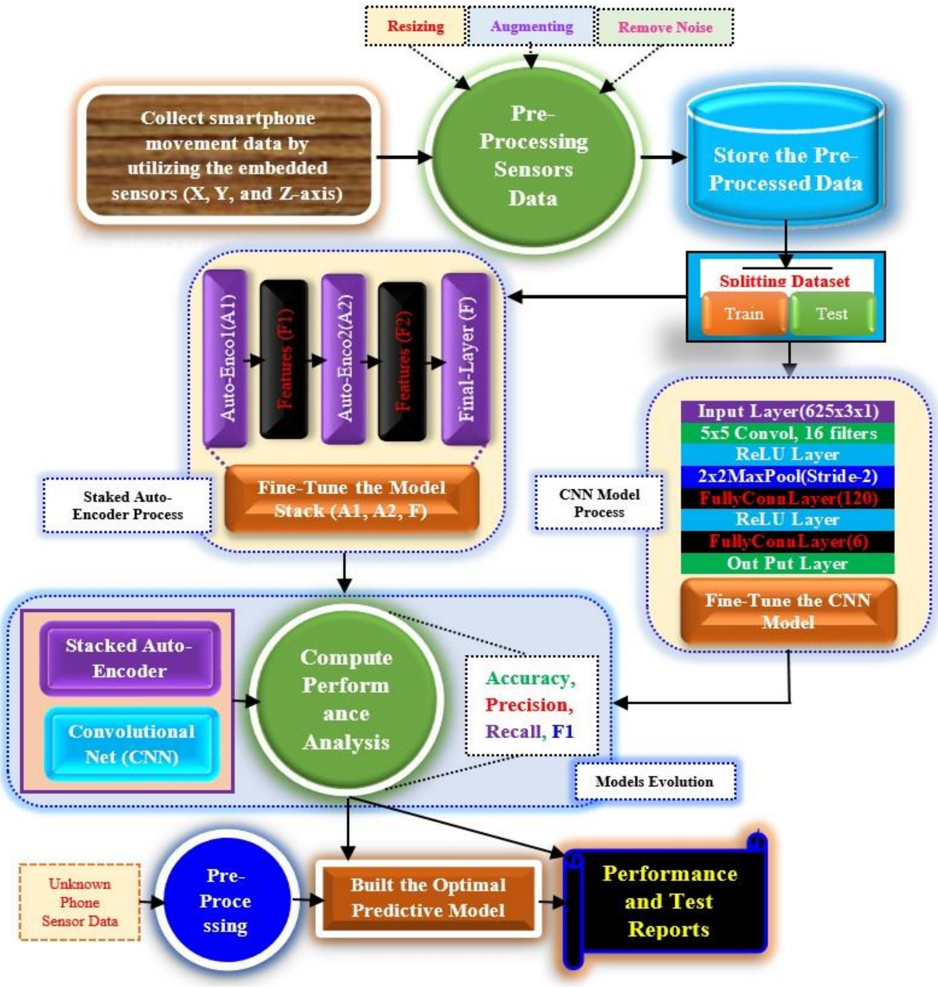


Figure 1. Proposal Model of Smart Phone Moment Identification System

A. *Proposal Model*

The proposed model in **Figure 1** is for a Smart Phone Moment Identification System. The research aims to collect movement data from smartphones using embedded sensors. The data is collected using embedded systems connected to a computer and stored for easy access. The data undergoes preprocessing steps to improve its quality. The preprocessed data is then divided into training and testing sets. Stacked Auto-Encoders and CNN models are used to learn patterns from the data. The models are fine-tuned to improve their performance. Finally, the models are compared and tested to accurately predict and classify smartphone movement patterns, which can be used in activity recognition and related applications.

B. Data Collection and Dataset Description

Figure 2 shows the SSD analysis's Data Collection, Expansion, and Data storage processes. Smartphones' embedded systems (ES) acquire data from built-in sensors like magnetometers, accelerometers, gyroscopes, and GPS receivers. These sensors continuously measure the smartphone's position, orientation, velocity, and acceleration. The ES transmits the acquired SD of the smartphone to the computer system. We can achieve this by using either wired or wireless communication protocols like USB, Bluetooth, or Wi-Fi. Computer systems like laptops, Desktops, or servers receive the SD transmitted by the smartphone's embedded system. It establishes a connection with the smartphone and prepares to process the received data. The computer system processes the received data from a sensor to derive meaningful information about the smartphone's movement. The computer system uses algorithms and software applications to interpret the SD and extract relevant motion-related parameters. The system analyses the smartphone's movement patterns using the processed SD. It can generate visualizations, graphs, or reports that provide insights into the smartphone's motion behavior over time. This analysis can be helpful for research, diagnostics, or monitoring purposes.

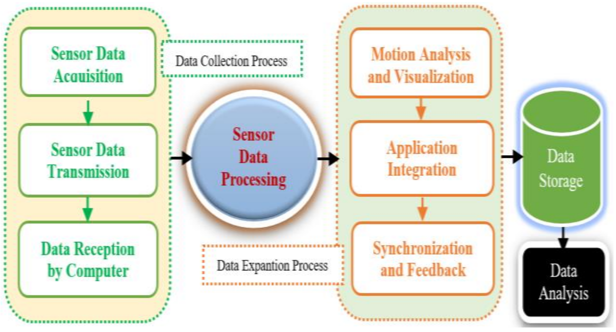


Figure 2. Proposal Model of Smart Phone Moment Identification System

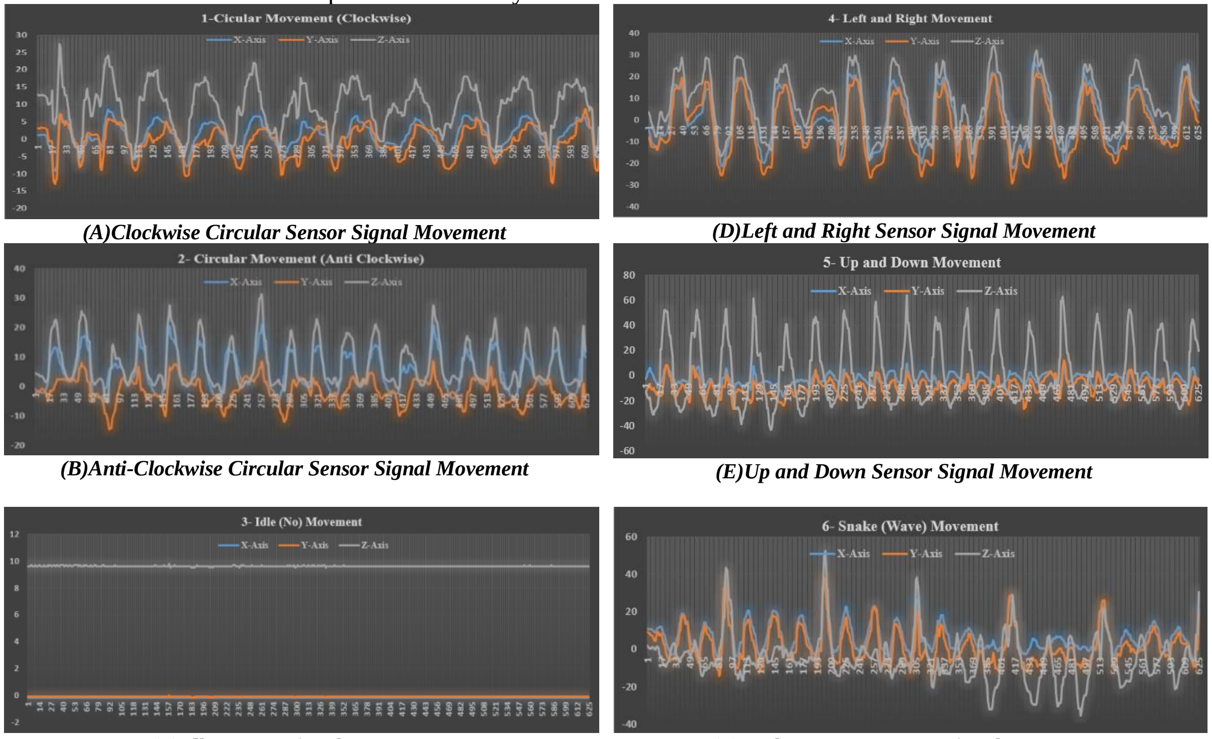


Figure 3. Smartphone Moment Signal all Clases

TABLE I. SMART PHONE SENSOR SIGNAL DATASET DESCRIPTION

Class (Code)	Smartphone Sensor Signal Form Description	Total Samples	Training Samples	Testing samples
CCM (1)	Clockwise Circular Sensor signal Movement	127	96	32
ACM (2)	Anti-Clockwise Circular Sensor signal Movement	92	72	20
IDM (3)	Idle or No Sensor signal Movement	167	127	40
LRM (4)	Left and Right Sensor signal Movement Right forms	177	132	45
UDM (5)	Up and Down Sensor signal Movement Idle forms	171	129	43
WAM (6)	Snake or Wave Sensor signal Movement	187	139	48
Total six Types of Movements	**Total Samples**	921	695	226

The computer system integrates the analyzed motion data with various applications or software systems. It can provide input to virtual or augmented reality applications, where the smartphone's movement influences the displayed content or virtual interactions. The computer system can synchronize the processed motion data with other devices or platforms. Based on the analyzed data, it can send feedback or commands back to the smartphone. For instance, it can trigger notifications, adjust settings, or control other connected devices based on the smartphone's movement. The computer system can store the processed motion data for further analysis or historical reference. It can perform advanced data analytics, pattern recognition, or ML techniques based on the data provided to gain insights or improve future motion-related algorithms. Figure 3 displays the six types of motions of the smartphone sensor signals in detail: clockwise motion (Figure 3 (A)), Anti-clockwise motion (Figure 3 (B)), Idle, and so on (Figure 3 (C, D, E, F)).

Table II describes signal form descriptions for different movement patterns, including clockwise circular, anti-clockwise circular, idle, left and proper, right forms, up and down, inactive forms, and snake or wave movement. The total samples for each available class code range from 82 to 160, indicating the overall size of the dataset for each specific movement pattern. Training samples provide necessary data to teach the model about the characteristics and patterns associated with each movement class, with varying numbers for each class. Testing samples analyze the task's performance and generalization ability of the trained models or algorithms, providing a benchmark for assessing how well the model can classify or predict movement patterns. The number of testing samples varies, ranging from 20 to 48 for different movement classes.

C. CNN Architecture

Figure 4 shows the user-defined CNN model. This simple CNN architecture takes a 625x3 as input and applies layers with complete connections, pooling, and convolution to learn and classify the input data into one of the six classes.

DataInputLayer: This layer specifies the input data shape as [625 3 1], demonstrating that the input consists of 625x3 sensor 3-axis data. Convolution2DLayer: This layer applies 16 filters of size 5x5 to the input image. The filters learn the incoming data's spatial properties.

ReLULayer: TheO/P of the layer that came before it is applied element-by-element by this layer using the Rectified Linear Unit (ReLU) activation function. ReLU introduces non-linearity to the network.

MaxPooling2DLayer: This layer performs max pooling with a stride of 2x2, reducing the spatial dimensions of the feature maps by a factor of 2. It helps in capturing the most salient features while reducing computational complexity.

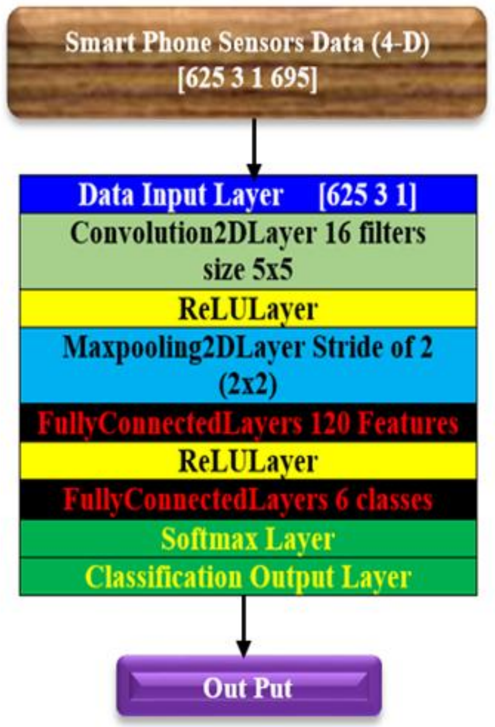

Smart Phone Sensors Data (4-D)
[625 3 1 695]
Data Input Layer [625 3 1]
Convolution2DLayer 16 filters size 5x5
ReLULayer
Maxpooling2DLayer Stride of 2 (2x2)
FullyConnectedLayers 120 Features
ReLULayer
FullyConnectedLayers 6 classes
Softmax Layer
Classification Output Layer
Out Put

Figure 4. 8-Layer CNN simple Model for Smart-Phone Sensor Data Analysis

FullyConnectedLayers: This layer consists of 120 neurons and is fully connected to the previous layer. It learns higher-level features by combining the extracted spatial features. This layer consists of 6 neurons, representing the number of classes in the classification task. It maps the learned features to the respective courses.

SoftmaxLayer: The softmax activation (SAF) function is used in this layer to normalize the previous layer'sO/Ps into probability values. It provides the final predicted probabilities for each class.

ClassificationOutputLayer: This layer computes the loss and performs the classification based on the predicted probabilities and the ground truth labels.

The I/P of the (8-layer) CNN model is the SSD in four dimensions. The total training samples are 695, with each sample size [625 3] having one channel. The first layer of the Data Input specifies the input data shape as [625 3 1], demonstrating that the input consists of 625x3 sensor 3-axis data. In the Convolution2D Layer, let X be the layer of I/P, combining Batch Size, Width, Height, and Channels. Let ‘W’ be the weight tensor of the shape of combinations of FilterWidth, Filter-Height, InputChannels, and O/P Channels. Let b represent the bias vector that shapes theO/P channels. The feature map Y is described as

$$Y[i,j,k] = ReLU\left(\sum(X[m,n,p] * W[i,j,p,k])\right) \quad (1)$$

Where ‘m’ in range (filter-width), for ‘n’ in range (filter-Height), for ‘p’ in range (input-channels, (input-channels)+

b[k]). ReLU Layer: Let X is I/P Layer. The O/P Y is computed as Eq. (2).

$$Y = Max(0, X) \tag{2}$$

Maxpooling2DLayer: Let X is layer of I/P, with shape (batch_size, width, height, channels). Let pool_size be the size of the pooling window (width, height). Let stride be the stride value for the pooling operation. The O/P feature map Y is calculated as Eq. (3).

$$Y[i,j,k] = Max(X[m,n,k]) \tag{3}$$

Where 'm' is in Range $(\text{istride}, \text{istride} + \text{pool\_size}[0])$ range for n in range (jstride, jstride + pool_size[1]))

Connected Layer: With shape (BatchSize, InputSize), let X serve as the input to the layer. Shape's weight matrix (InputSize, OutputSize) can be represented by W. Shape's bias vector (OutputSize), let b be. The result, Y, is calculated as

$$Y[i,j,k] = ReLU(X * W + b) \tag{4}$$

ReLU Layer: Let X serve as the layer's input. The result, Y, is calculated as

$$Y = Max\ (0,\ X) \tag{5}$$

Fully Connected Layer: With shape (BatchSize, InputSize), let X serve as the input to the layer. The O/P Y is calculated as W is the shape's weight matrix (InputSize, OutputSize), and b is its bias vector (OutputSize).

$$Y = X * W + b \tag{6}$$

Softmax Layer: With shape (BatchSize, InputSize), let X serve as the input to the layer. The result, Y, is calculated as

$$Y[i,j] = \frac{e^{(X[i,j])}}{\Sigma(e^{(X[i,k])})} \tag{7}$$

Where k in range (1, input_size))

D. Staked Auto-Encoder

Figure 5 shows the block diagram of the various layers of the 2-Layer Stacked Auto-encoder, where each layer consists of an encoder and a decoder, and softmax layers. The Input (I/P L) Layer represents the I/P data that flows into the auto-encoder.

The Layer 1 Encoder that this block performs a linear transformation and applies a non-linear activation function to compress the I/P data into a lower-dimensional representation. The first auto-encoder compresses the input, capturing its essential features and reducing its dimensionality.

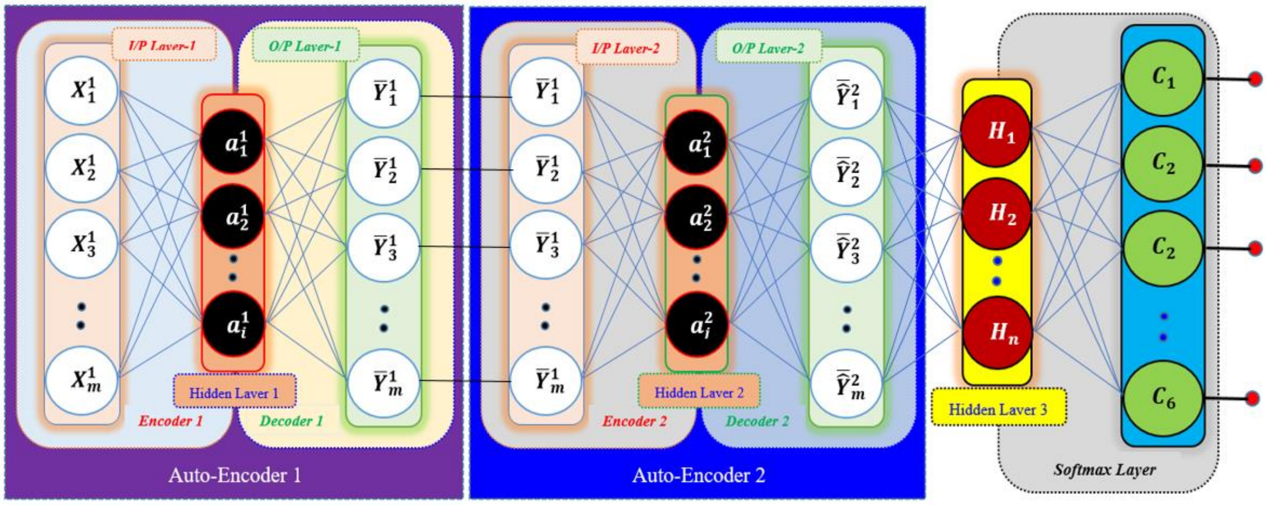


Figure 5. 2-Layer Stacked Auto-Encoder Model for Smart-Phone Sensor Data Analysis

The second auto-encoder further squeezes the O/Ps from Auto-Encoder 1, extracting higher-level representations of I/P Data. The Layer 2 Decoder receives the compressed representation from Layer 2's encoder and reconstructs the data by applying reverse transformation and activation. The softmax layer takes the O/P from Layer 2's decoder and performs from and uses the softmax (SMF) function, which makes them probabilities for each class. The O/P layer of the model provides the final classification result, identifying the most probable class for a given input based on the probabilities obtained from the softmax layer [34].

Layer1: Let X is the Dataset with elements set {X1, X2… Xm} then the Auto-Encoder 1's encoder denotes Eq. (1) and computation as follows as

$$E^{(1)} = f\left(X * W^{(1)} + b^{(1)}\right) \tag{8}$$

where $W^{(1)}$ is the weight matrix, $b^{(1)}$ is the bias vector, and f represents the activation function for the encoder of the first layer.

The Auto-Encoder 1's encoder denotes $D^{(1)}$ and computation as follows as

$$D^{(1)} = f\left(E^{(1)} * W'^{(1)} + b^{(1)}\right) \tag{9}$$

Here, $E^{(1)}$ is the encoded representation, $W'^{(1)}$ is the transpose of the weight matrix, $b^{(1)}$) is the bias vector, and f represents the activation function for the decoder of the first layer.

Layer 2: The Auto-Encoder-2's encoder denotes E(2) and computation as follows as

$$E^{(2)} = f\left(E^{(1)} * W^{(2)} + b^{(2)}\right) \qquad (10)$$

where $W^{(2)}$ is the weight matrix, $b^{(2)}$ is the bias vector, and f represents the activation function for the encoder of the first layer. The Auto-Encoder-2's encoder denotes D(2) and computation as follows as

$$D^{(2)} = f\left(E^{(2)} * W'^{(2)} + b^{(2)}\right) \qquad (11)$$

Here, $E^{(2)}$ is the encoded representation, $W'^{(2)}$ is the transpose of the weight matrix, $b^{(2)}$) is the bias vector, and f represents the activation function for the decoder of the first layer.

Softmax Layer: Assuming the input to the softmax layer is denoted as Z, that is O/P of Auto-Encoder-2, and it has dimensionality (batch_size, num_classes), where batch_size is the number of samples and num_classes is the number of classes in the classification task. The softmax function takes the input Z and computes the probabilities for each class. The probabilities are calculated by exponentiation of the input values and normalizing them across all classes. The softmax function can be defined as

$$P(Class\ i) = \frac{e^{(Z[:,i])}}{\sum(e^{(Z[:,i])},\ axis=1)} \qquad (12)$$

Here, P(class i) represents the probability of class i, Z[: i] denotes the values of the ith class in the input Z, and sum(exp(Z), axis=1) calculates the sum of exponentiated values across all classes.

In classification tasks, a common loss function used with softmax is the cross-entropy loss. Given the predicted probabilities P and the true labels Y, the cross-entropy loss can be calculated as

$$L = -\sum(Y * \log(P)\,, axis = 1) \qquad (13)$$

Here, Y represents the one-hot encoded true labels, log(P) denotes the element-wise logarithm of the predicted probabilities, and the sum is taken across the classes (axis=1).

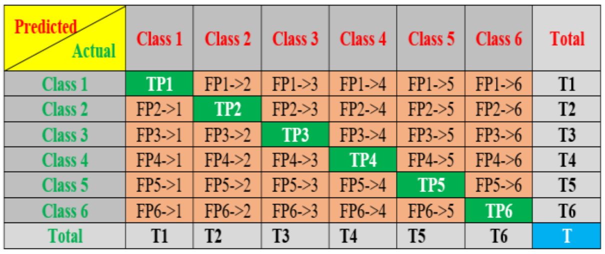
The softmax layer is typically used in the O/P layer of a neural network for multi-class classification tasks. The overall objective of the stacked auto-encoder combines the pre-training and fine-tuning objectives.

$$L\_total = \begin{pmatrix} \left(\lambda^1 * \mathrm{L}^1_{\mathrm{pretrain}}\right) + \\ \left(\lambda^2 * \mathrm{L}^2_{\mathrm{pretrain}}\right) + \\ (\lambda\_\mathrm{task} * \mathrm{L\_task}) \end{pmatrix} \qquad (14)$$

Here, λ_1, λ_2, and λ_task are weighting factors that control the importance of each component in the overall objective. L_1_pretrain and L_2_pretrain represent the reconstruction losses for the pre-training of the respective layers, and L_task represents the loss function specific to the task being performed.

E. Confusion Matrix and Performance Parameters

In **Figure 6**, a confusion matrix represents a problem with six classes. The matrix uses arrows to show when a class is misclassified as another class. For example, if an arrow from FP1 to 2 indicates that instances of Class 1 were misclassified as Class 2.



Predicted / Actual	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Total
Class 1	TP1	FP1->2	FP1->3	FP1->4	FP1->5	FP1->6	T1
Class 2	FP2->1	TP2	FP2->3	FP2->4	FP2->5	FP2->6	T2
Class 3	FP3->1	FP3->2	TP3	FP3->4	FP3->5	FP3->6	T3
Class 4	FP4->1	FP4->2	FP4->3	TP4	FP4->5	FP4->6	T4
Class 5	FP5->1	FP5->2	FP5->3	FP5->4	TP5	FP5->6	T5
Class 6	FP6->1	FP6->2	FP6->3	FP6->4	FP6->5	TP6	T6
Total	T1	T2	T3	T4	T5	T6	T

Figure 6. General Confusion Matrix for 6-classes

AUC is a performance metric used in ML to evaluate the quality classification model. It is calculated by computing the area under the receiver operating characteristic (ROC) curve. The ROC curve is a plot of true (TPR) positive rate against false (FPR) positive rate for different classification thresholds [35] [36].

CA (Classification Accuracy) is the ratio of correctly classified samples to the total number of samples in a dataset. It is a basic performance metric used to evaluate the overall accuracy of a classification model. The calculations as follows [37].

$$CA = \frac{(\text{No.of correctly classified instances})}{(\text{Total number of instances})} \qquad (15)$$

F1 score is recall and precision's harmonic means, two other important performance metrics for classification models. F1 score takes into account both recall and precision and provides a balanced evaluation of the model's performance [38]. It is calculated as

$$F1 = 2 * \left(\frac{(\text{precision} * \text{recall})}{(\text{precision}+\text{recall})}\right) \qquad (16)$$

Precision is the TP ratio to the sum of TP and FP instances. It measures how many of the positive instances model predictors are true positive instances.

$$\text{Precision} = \left(\frac{(\text{TP})}{(\text{TP+FP})}\right) \qquad (17)$$

Recall is the ratio of true positive (TP) to the sum of TP and false negative (FN) instances. It measures how many of the actual positive instances the model has correctly identified.

$$\text{Precision} = \left(\frac{(\text{TP})}{(\text{TP+FN})}\right) \qquad (18)$$

4. Result Analysis

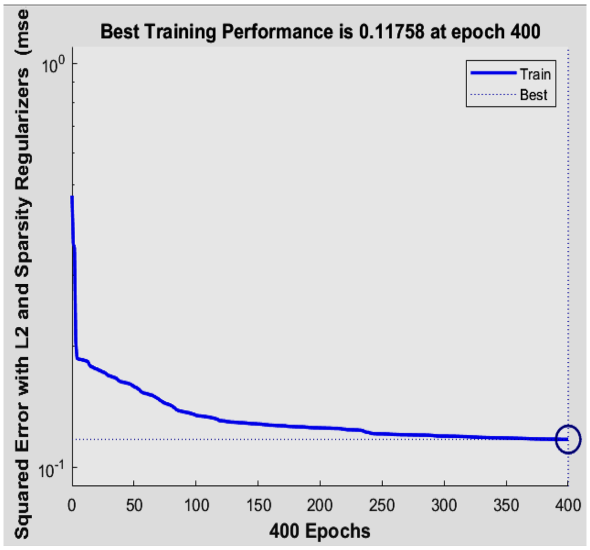
This section shows that the embedded DL models identify smartphone movements well. These models use DL algorithms on the smartphone, allowing them to identify real-time movements. This makes them useful for things like fitness tracking and augmented reality. The analysis of the results shows that embedded DL models have a lot of potential and advantages for identifying smartphone movements. This research helps advance the field of intelligent smartphone applications and can inspire future research.

A. Two-Layer Stacking Auto-Encoder Results Analysis

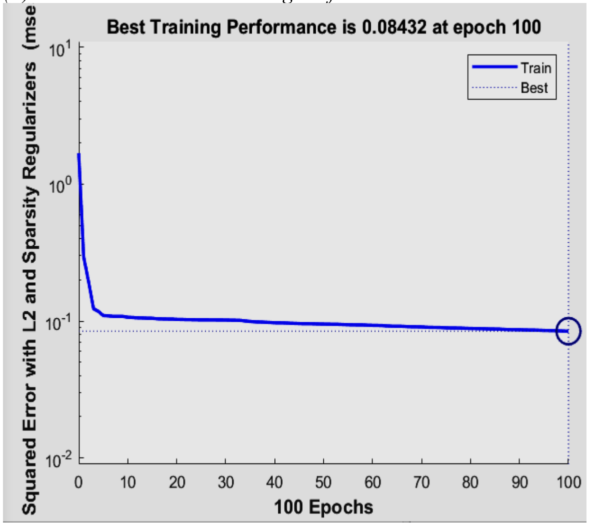
Load the data from the 'WaveTrain.mat' and 'WaveTest.mat' files using the load() function. Assign the loaded data to the appropriate variables: trainData, trainLabels, testData, and test Labels. Preprocess the training and testing data by dividing each element of the cell array by the maximum absolute value of the array using the cell fun() function. Set the parameters for the autoencoder1. Define hiddenSize1 as 100, MaxEpochs as 400, L2WeightRegularization as 0.004, SparsityRegularization as 4, SparsityProportion as 0.15, and ScaleData as false. Train the first autoencoder using the trainAutoencoder() function with train data and the defined parameters. Store Trained_Autoencoder in autoenc1. Encode the training data using autoenc1 and store the result in feat1. Set the parameters for the autoencoder2. Define hiddenSize2 as 50, MaxEpochs as 100, L2WeightRegularization as 0.002, SparsityRegularization as 4, SparsityProportion as 0.1, and ScaleData as false.

Train the second autoencoder using the trainAutoencoder() function with feat1 and the defined parameters. Store TrainedAutoencoder in autoenc2. Encode the features feat1 using the encode method of autoenc2 and save the resulting features as feat2. Train the softmax layer with the TrainSoftmaxLayer function using the encoded features feat2 and the training labels trainable with the specified hyperparameters MaxEpochs. Save the resulting softmax layer as softnet. Stack the three trained neural networks (autoenc1, autoenc2, and softnet) together using the stack function and save the resulting stacked neural network as stackednet. Convert the training data TrainData into a matrix xTrain with the size inputSize (the number of features) by numeral (TrainData) (the number of training samples) by flattening each training data element. Fine-tune the stacked neural network stackednet with the training matrix xTrain and corresponding training labels trainLabels using the train function.

Figure 7 shows the Best Training Performance (MSE) analysis of Auto-Encoder 1 and Auto-Encoder 2 in the Staking Auto-Encoder model. In this, Figure 7(A) shows the best performance analysis with a 0.11758 MSE value at epoch 400 of Auto-Encoder 1. Figure 7(B) shows the best performance analysis with the 0.08432 MSE value of Auto-Encoder 2 at epoch 100.



(A) Auto-Encoder 1 Best Training Performance

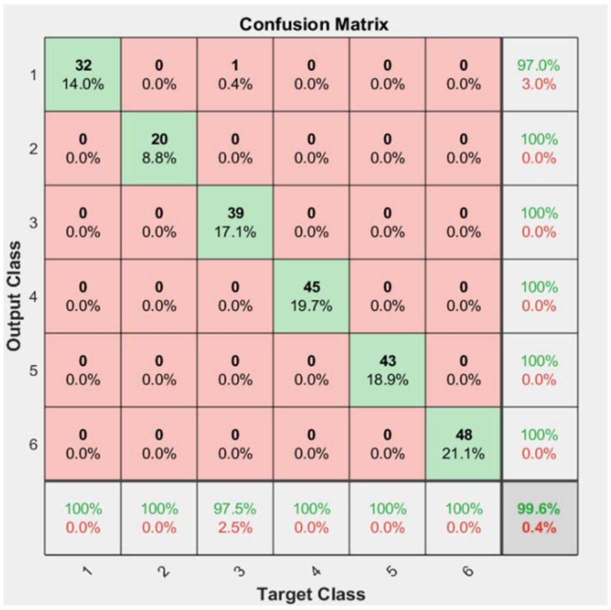


(B) Auto-Encoder 2 Best Training Performance

Figure 7. Best Training Performance (MSE) Analysis

Figure 8 shows the confusion matrix of the Stacking Auto-Encoder for the testing dataset. As per the Confusion matrix analysis, most of the five classes performed well with 100% accuracy. In class 3, one miss calculated value is classified as class 1. The total accuracy of the training dataset is 99.8%, and the testing dataset is 99.6%.

The performance of the model was evaluated using different parameters. **Table III** shows the detailed analysis of the performance metrics. The accuracy of correctly classifying instances for most classes was perfect, with a score of 1.00. However, class 3 had a slightly lower accuracy of 98.5%. The model also had high AUC values, indicating its ability to distinguish between different classes accurately. The F1 scores, which measure the balance between precision and recall, were very high for all classes. Most classes had excellent precision values, indicating zero false positives, and recall values, indicating no false negatives. Overall, the stacked autoencoder model performed excellently across all classes, with high AUC values, accurate predictions, and balanced precision and recall. The average performance measures further confirmed the model's effectiveness, with high values for AUC, accuracy, F1-score, precision, and recall. These results demonstrate that the model can accurately identify and classify different movement types.



Confusion Matrix

Output Class \ Target Class	1	2	3	4	5	6	
1	32 14.0%	0 0.0%	1 0.4%	0 0.0%	0 0.0%	0 0.0%	97.0% 3.0%
2	0 0.0%	20 8.8%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
3	0 0.0%	0 0.0%	39 17.1%	0 0.0%	0 0.0%	0 0.0%	100% 0.0%
4	0 0.0%	0 0.0%	0 0.0%	45 19.7%	0 0.0%	0 0.0%	100% 0.0%
5	0 0.0%	0 0.0%	0 0.0%	0 0.0%	43 18.9%	0 0.0%	100% 0.0%
6	0 0.0%	0 0.0%	0 0.0%	0 0.0%	0 0.0%	48 21.1%	100% 0.0%
	100% 0.0%	100% 0.0%	97.5% 2.5%	100% 0.0%	100% 0.0%	100% 0.0%	99.6% 0.4%

Figure 8. General Confusion Matrix for 6-classes

TABLE III
PERFORMANCE PARAMETERS OF STACKING AUTO-ENCODER

Class(Ind.)	AUC	CA	F1	Precision	Recall
CCM (1)	1.00	1.00	0.985	1.000	0.970
ACM (2)	1.00	1.00	1.000	1.000	1.000
IDM (3)	0.99	0.98	0.987	0.975	1.000
LRM (4)	1.00	1.00	1.000	1.000	1.000
UDM (5)	1.00	1.00	1.000	1.000	1.000
WAM (6)	1.00	1.00	1.000	1.000	1.000
Overall	**1.00**	**0.996**	**0.9953**	**0.9958**	**0.995**

B. Eight-Layer Convolution Neural Network (CNN) Results Analysis

The Smart Phone movement SD Set Training Progress is shown in Figure 4. The CNN architecture consists of eight layers. This architecture is commonly used to classify SSD. The input to the network is sensor signal data with a size of 625x3x1. The final layer uses the softmax activation function to classify the data into six classes. The InputLayer defines the input size as [625 3 1], indicating the size of the sensor signal data. The convolution2dLayer performs convolutional operations on the input using 16 filters of size 5x5. The 'Padding' parameter is set to 'same' to maintain the input size. The ReLULayer applies the ReLU activation function to introduce non-linearity. The maxPooling2dLayer reduces the spatial dimensions of the previous layer's O/P by a factor of 2 using a 2x2 window and a stride of 2. The FullyConnectedLayer has 120 output neurons and receives flattened input from the previous layer. Another ReLU activation layer is applied after the FullyConnectedLayers.

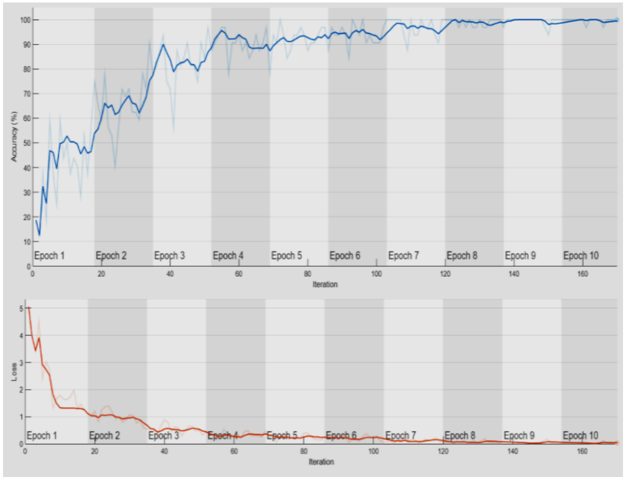
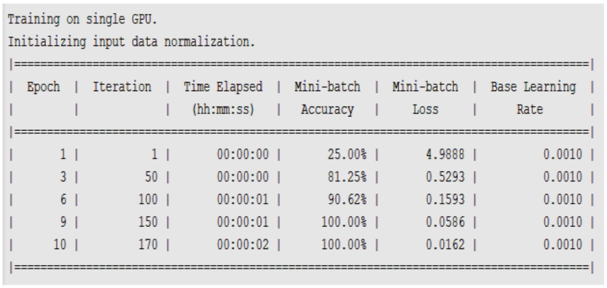


Figure 9. CNN Smart Phone movement Sensor Data Set Training Progress

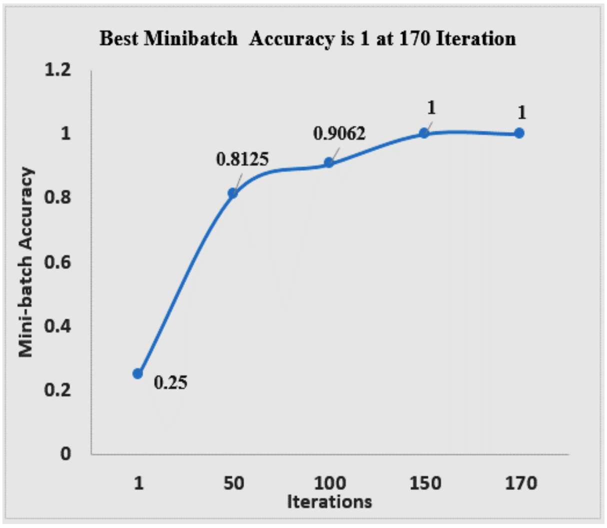
The FullyConnectedLayers (FCL) is a layer with six O/P neurons representing measurements or members of different classes in a classification task. The SoftMax Layer (SML) uses the (SAF) softmax activation function to generate a probability distribution across the classes. The final layer, called ClassificationLayer, calculates the cross-entropy loss between the actual and predicted label probabilities. **Figure 9** shows the progress of the Smart Phone Movement SD Set training. The blue line represents the training process, while the red line shows the loss values. The training dataset is processed in 4 seconds using the mini-batch loss function. The training process utilizes a hardware GPU. The light blue line indicates the training accuracy for each iteration, while the dark blue line represents the smoothed training accuracy points. The light red line shows the training loss value for each iteration.

Figure 10 shows the Best Training Process with Minibatch Accuracy and Loss Values. The input describes the training process of a CNN using mini batches. The training process is divided into epochs, each consisting of multiple iterations. In the first epoch, at the first iteration, the mini-batch accuracy is 25.00%, and the mini-batch loss is 4.9888. The training time elapsed is 00:00:00. After training for three epochs, at the 50th iteration of the third epoch, the mini-batch accuracy has improved to 81.25%, and the mini-batch loss has reduced to 0.5293. The training time elapsed is still 00:00:00. At the 100th iteration of the sixth epoch, the mini-batch accuracy has increased to 90.62%, and the mini-batch loss has significantly reduced to 0.1593. The training time has elapsed now 00:00:01. Finally, at the last epoch, the training is completed at the 170th iteration (Epoch 10, Iteration 170). The mini-batch accuracy remains 100.00%, reducing the loss to 0.0162. The total training time elapsed is 4 seconds.

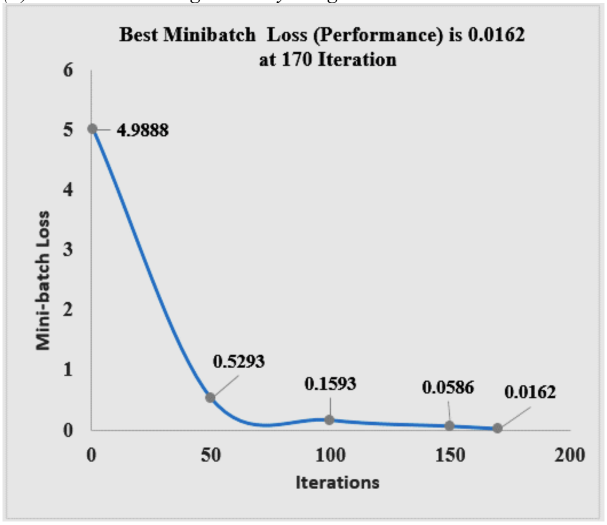


```
Training on single GPU.
Initializing input data normalization.
|======================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|======================================================================================|
|       1 |           1 |       00:00:00 |       25.00% |       4.9888 |          0.0010 |
|       3 |          50 |       00:00:00 |       81.25% |       0.5293 |          0.0010 |
|       6 |         100 |       00:00:01 |       90.62% |       0.1593 |          0.0010 |
|       9 |         150 |       00:00:01 |      100.00% |       0.0586 |          0.0010 |
|      10 |         170 |       00:00:02 |      100.00% |       0.0162 |          0.0010 |
|======================================================================================|
```

Figure 10. Best Training Process Step by Step with Time Elapsed and Base Learning Rate



(A) CNN-Best Training Accuracy using Mini-batch



(B) CNN Best- Training performance using Mini-Batch(Loss)

Figure 11. Best Training Process with Minibatch Accuracy and Loss Values

Figure 11 shows the Best Training Process with Minibatch Accuracy and Loss Values. Figure 11 (A) shows the analysis of CNN's best mini-batch accuracy. It achieved one at 170 iterations (10 epochs). Figure 11 (B) indicates the study of CNN's best mini-batch loss value.

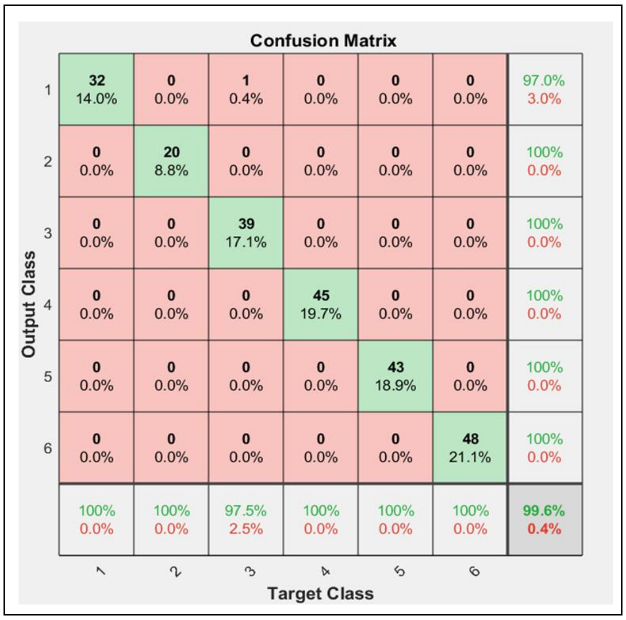


Figure 12. CNN Confusion Matrix for the Testing Dataset

TABLE IV
PERFORMANCE PARAMETERS OF THE CNN MODEL

Class	AUC	CA	F1	Precision	Recall
CCM (1)	0.991	0.969	0.984	0.969	1.000
ACM (2)	1.00	1.00	1.000	1.000	1.000
IDM (3)	1.00	1.00	1.000	1.000	1.000
LRM (4)	1.00	1.00	1.000	1.000	1.000
UDM (5)	0.997	0.977	0.988	0.977	1.000
WAM (6)	1.00	1.00	0.978	1.000	0.960
Overall	**0.998**	**0.991**	**0.992**	**0.991**	**0.993**

Figure 12 shows the CNN confusion matrix for the Testing Dataset and analysis. **Table IV** shows the performance parameters of the Stacking Auto-Encoder. The class CCM demonstrates an AUC value of 0.991, indicating excellent performance in distinguishing positive and negative instances. The classification accuracy is 0.969, suggesting that 96.9% of cases were correctly classified. The F1 score, which combines precision and recall, is 0.984, indicating a successful balance between recall and precision. The precision is 0.969, representing the symmetry of all instances that were correctly predicted to be positive. The recall is 1.000, demonstrating that all positive instances were correctly identified. The Class ACM demonstrates outstanding performance in all measures. The AUC value of 1.000 indicates perfect discrimination between positive and negative instances. The classification accuracy of 1.000 means that all the cases were correctly classified. The F1 score, precision, and recall are all 1.000, suggesting flawless performance in predicting positive instances. Class 2, classes 3, and 4 perform well. Class UDM achieves high scores across all performance measures. The AUC value of 0.997 indicates excellent discrimination ability. The classification accuracy is 0.977, suggesting that 97.7% of instances were correctly classified.

The F1 value is 0.988, showing a good balance. The precision is 0.977, and the recall is 1.000 with good balance. Class WAM demonstrates high performance in most measures. The AUC value of 1.000 indicates perfect discrimination ability. The classification accuracy is 1.000, meaning that all instances were correctly classified. The F1 score is 0.978, precision is 1.000, and recall is 0.960, indicating a good balance dataset for the model. The overall performance of the classification model is excellent, with an AUC value of 0.998, indicating high discrimination ability. The classification accuracy is 0.991, suggesting that 99.1% of instances were correctly classified. The F1 score of 0.992 reflects a good balance of precision of 0.991 and recall of 0.993.

5. DISCUSSIONS

This research discusses the advantages of using embedded DL models to identify smartphone movements. These models can use DL algorithms directly on the smartphone without relying on cloud-based processing. The research compares these embedded models to similar studies and discusses the trade-offs in complexity and computational requirements. The smartphone can identify movements in real time using embedded DL models. This is useful for applications like fitness tracking, health monitoring, and augmented reality, where immediate feedback is essential. The research shows that both 2-layer Stacked Auto-Encoders and 8-layer CNNs perform well in accurately classifying smartphone movements.

Both models in this analysis have achieved perfect accuracy in classifying smartphone movements. The 2-layer Stacked Auto-Encoders performed better than the CNN model, with higher values for all classes and overall AUC and CA. It indicates that the Stacked Auto-Encoders are more effective and reliable in accurately classifying smartphone movements. Other researchers have also proposed models for DL with different datasets, but they have achieved moderate accuracy compared to the proposed model in this analysis. Peppas et al. [39] suggested a real-time CNN model that effectively recognizes human physical activity using accelerometer data on a mobile device. It achieved a high classification accuracy of 94.18% on the WISDM dataset and 79.12% on the Actitracker dataset. Thakur et al. [40] proposed a unified architecture called "ConvAE-LSTM" that combines CNNs, auto-encoders, and LSTM networks for smartphone-based activity recognition. Their model achieved a remarkable accuracy of 95.69% on the OPPORTUNITY dataset, surpassing the accuracy of Random Forest and Support Vector Machine models. Khan et al. [41] conducted experiments and collected a comprehensive dataset of nine daily activities. They trained various ML models on SD from smartphones and wearable devices. The random forest algorithm achieved a test accuracy of 95%, while a custom-built Bi-LSTM model outperformed it with an improved accuracy of 98.1%. **Table V** compares the proposed model with other research models in the analysis.

TABLE V
SOME OF THE RESEARCH WORKS COMPARED WITH PRESENT RESEARCH AND ANALYSIS

Ref. No.	Author & Year	Description	Results and Models Analysis
[42]	Straczkiewicz et al.	Using the UniMiB-SHAR public dataset, the authors proposed a DL method for HAR that makes use of the Resnet architecture.	The rigorous leave-one-subject evaluation yielded significant results, with accuracy rising from **78.24% to 80.09 %** and the F1-score rising from **78.40 % to 79.36 %**.
[43]	Ignatov et al.	The proposed method utilizes CNNs to achieve real-time HAR from smartphones' accelerometers.	CNN + stat. features + data centring **97.63%** of accuracy
[44]	Mutegeki et al.	In this, the authors introduced a holistic DL-based HAR architecture, CNN-LSTM, which enhances predictive accuracy, reduces model complexity, and eliminates the need for advanced feature engineering by leveraging both spatial and temporal depth within the network.	The proposed model achieved exceptional accuracy, with **99%** on the iSPL dataset and **92%** on the UCI HAR public dataset, outperforming other DNN architectures and ML models.
[45]	Huang et al.	A novel method for automatically extracting activity recognition features from signals from mobile sensors that capture Scale invariance and local dependence was presented by the authors.	The CNN-based model achieves higher accuracy than the best algorithm (PCA-ECDF) with confidence levels of **95%**, with classification accuracies of **88.19%**.
[46]	Rahman et al.	The approach employs CNNs for real-time HAR from smartphones' accelerometers data.	CNN model achieved high Accuracy of **85.1%**
[47]	de et al.	Authors introduced the potential of using CNNs and RNNs (LSTM and GRU) for activity identification from SSD, achieving improved accuracy compared to previous approaches on the UniMiB SHAR dataset.	Classifying 17 types of activities, using a 5-fold cross-validation method, was **95.49%** with a GRU network.
[48]	Aquino et al.	By providing explanations for their decision-making process, this study aims to improve the interpretability of 1-D CNN models in HAR application.	1-D CNN achieved an accuracy of **0.978 and 0.755**, utilizing SD and signal intensity techniques, and the BUI network achieved an average accuracy of **0.937**.
[49]	Gamble et al.	The authors introduced a DL approach using 1D-CNN model for HAR and HARI by extracting features from smartphone accelerometer and gyroscope signals, aiming to identify both the activity and the identity of the entity carrying the smartphone.	the 1D-CNN model achieved a high accuracy of **96.77%** in classifying activities and **82.37%** in classifying identity within a one-second time window using the Motion Sense dataset.

[50]	Aquino et al.	The authors introduced a novel XAI method that utilizes t-SNE learned features in one-dimensional CNNs. It showcased performance on two public databases SHO and HAPT.	On the SHO dataset, CNN models achieve **0.98** accuracy, while on the HAPT dataset, they achieve **0.93** accuracy.
	Present study	**Experiment on synthesis (real-world) data set**	**8-layer CNN-Model AUC -0.998 Accuracy-99.1%**
	Present study	**Experiment on synthesis (real-world) data set**	**2-layer Auto-Encoder AUC-1.0 Accuracy-99.6%**

6. CONCLUSION

This study shows that using DLmodels like CNN and SAEs can accurately identify different types of smartphone movements. The SAEs performed well during training, with Auto-Encoder 1 and Auto-Encoder 2 achieving the best results at epochs 400 and 100, respectively. The SAEs also performed well regarding AUC, CA, F1-score, precision, and recall, meaning they can accurately identify and classify smartphone movements. The CNN model also performed well, showing high values for AUC, CA, F1-score, precision, and recall. The study indicates that CNNs can capture complex patterns and features from the input data. Implementing these DL models for smartphone movement identification can have real-time applications in fitness tracking, health monitoring, and augmented reality. The researchers found that using models directly on smartphones can identify movement in real-time without relying on cloud-based processing. These models, such as CNNs and SAEs, are accurate and perform well, which can improve smartphone applications. Future researchers may explore more advanced DL methods like RNNs or TLs to improve accuracy. The goal is to make the system practical and valuable for different applications. Future work may involve developing techniques that allow the models to adapt to other users, activities, and smartphone sensors. The researchers will continue to enhance the models and optimize their performance.

REFERENCES

[1] Zhang, H., Yan, X., Li, H., Jin, R., & Fu, H. (2019). Real-time alarming, monitoring, and locating for non-hard-hat use in construction. Journal of construction engineering and management, 145(3), 04019006.DOI: https://doi.org/10.1061/(ASCE)CO.1943-7862.0001629

[2] Diraco, G., Rescio, G., Siciliano, P., & Leone, A. (2023). Review on Human Action Recognition in Smart Living: Sensing Technology, Multimodality, Real-Time Processing, Interoperability, and Resource-Constrained Processing. Sensors, 23(11), 5281. https://doi.org/10.3390/s23115281

[3] Ghorbani, Fatemeh, Amirmasoud Ahmadi, Mohammad Kia, Quazi Rahman, and Mehdi Delrobaei. "A decision-aware ambient assisted living system with IoT embedded device for in-home monitoring of older adults." Sensors 23, no. 5 (2023): 2673. https://doi.org/10.3390/s23052673

[4] Beg, S., Handa, M., Shukla, R., Rahman, M., Almalki, W. H., Afzal, O., & Altamimi, A. S. A. (2022). Wearable smart devices in cancer diagnosis and remote clinical trial monitoring: Transforming the healthcare applications. Drug Discovery Today. https://doi.org/10.1016/j.drudis.2022.06.014

[5] Huang, X., Li, Y., Chen, J., Liu, J., Wang, R., Xu, X., ... & Guo, J. (2019). Smartphone-based blood lipid data acquisition for cardiovascular disease management in internet of medical things. IEEE access, 7, 75276-75283. DOI: 10.1109/ACCESS.2019.2922059

[6] Johnson, G., & Rajamani, R. (2019). Smartphone localization inside a moving car for prevention of distracted driving. Vehicle system dynamics. DOI: https://doi.org/10.1080/00423114.2019.1578889

[7] Masud, M. T., Mamun, M. A., Thapa, K., Lee, D. H., Griffiths, M. D., & Yang, S. H. (2020). Unobtrusive monitoring of behavior and movement patterns to detect clinical depression severity level via smartphone. Journal of biomedical informatics, 103, 103371.DOI: https://doi.org/10.1016/j.jbi.2019.103371

[8] Qi, W., Wang, N., Su, H., & Aliverti, A. (2022). DCNN based human activity recognition framework with depth vision guiding. Neurocomputing, 486, 261-271. doi: https://doi.org/10.1016/j.neucom.2021.11.044

[9] Wang, J., Chen, Y., Hao, S., Peng, X., & Hu, L. (2019). Deep learning for sensor-based activity recognition: A survey. Pattern recognition letters, 119, 3-11. DOI: 10.1016/j.patrec.2018.02.010

[10] Arikumar, K. S., Prathiba, S. B., Alazab, M., Gadekallu, T. R., Pandya, S., Khan, J. M., & Moorthy, R. S. (2022). FL-PMI: federated learning-based person movement identification through wearable devices in smart healthcare systems. Sensors, 22(4), 1377. DOI: https://doi.org/10.3390/s22041377

[11] Liu, P., Yang, P., Song, W. Z., Yan, Y., & Li, X. Y. (2019, April). Real-time identification of rogue WiFi connections using environment-independent physical features. In IEEE INFOCOM 2019-IEEE Conference on Computer Communications (pp. 190-198). IEEE. DOI: 10.1109/INFOCOM.2019.8737455

[12] Lu, L., Yu, J., Chen, Y., Liu, H., Zhu, Y., Kong, L., & Li, M. (2019). Lip reading-based user authentication through acoustic sensing on smartphones. IEEE/ACM Transactions on Networking, 27(1), 447-460. DOI: 10.1109/TNET.2019.2891733

[13] Jiang, X., Lu, Y., Lu, Z., & Zhou, H. (2018). Smartphone-based human activity recognition using CNN in frequency domain. In Web and Big Data: APWeb-WAIM 2018 International Workshops: MWDA, BAH, KGMA, DMMOOC, DS, Macau, China, July 23–25, 2018, Revised Selected Papers 2 (pp. 101-110). Springer International Publishing. DOI: https://doi.org/10.1007/978-3-030-01298-4_10

[14] Zhou, B., Yang, J., & Li, Q. (2019). Smartphone-based activity recognition for indoor localization using a convolutional neural network. Sensors, 19(3), 621. doi:10.3390/s19030621

[15] Ravi, D., Wong, C., Lo, B., & Yang, G. Z. (2016). A deep learning approach to on-node sensor data analytics for mobile or wearable devices. IEEE journal of biomedical and health informatics, 21(1), DOI: 56-64.10.1109/JBHI.2016.2633287

[16] Stampfler, T., Elgendi, M., Fletcher, R. R., & Menon, C. (2023). The use of deep learning for smartphone-based human activity recognition. Frontiers in Public Health, 11. doi: 10.3389/fpubh.2023.1086671

[17] Qi, W., Su, H., Yang, C., Ferrigno, G., De Momi, E., & Aliverti, A. (2019). A fast and robust deep convolutional neural networks for complex human activity recognition using smartphone. Sensors, 19(17), 3731. doi:10.3390/s19173731

[18] Xia, K., Huang, J., & Wang, H. (2020). LSTM-CNN architecture for human activity recognition. IEEE Access, 8, 56855-56866. doi: 10.1109/ACCESS.2020.2982225

[19] Ye, W., Cheng, J., Yang, F., & Xu, Y. (2019). Two-stream convolutional network for improving activity recognition using convolutional long short-term memory networks. IEEE Access, 7, 67772-67780. doi: 10.1109/ACCESS.2019.2918808

[20] Ponciano, V., Pires, I. M., Ribeiro, F. R., Garcia, N. M., Pombo, N., Spinsante, S., & Crisóstomo, R. (2019, September). Smartphone-based automatic measurement of the results of the Timed-Up and Go test. In Proceedings of the 5th EAI International Conference on Smart Objects and Technologies for Social Good (pp. 239-242). DOI: https://doi.org/10.1145/3342428.3343035

[21] Qin, Z., Zhang, Y., Meng, S., Qin, Z., & Choo, K. K. R. (2020). Imaging and fusing time series for wearable sensor-based human activity recognition. Information Fusion, 53, 80-87. DOI: https://doi.org/10.1016/j.inffus.2019.06.014

[22] Dasgupta, N., Funk, M. J., Lazard, A., White, B. E., & Marshall, S. W. (2020). Quantifying the social distancing privilege gap: a longitudinal study of smartphone movement. MedRxiv, 2020-05. DOI: https://doi.org/10.1101/2020.05.03.20084624

[23] Alo, U. R., Nweke, H. F., Teh, Y. W., & Murtaza, G. (2020). Smartphone motion sensor-based complex human activity identification using deep stacked autoencoder algorithm for enhanced smart healthcare system. Sensors, 20(21), 6300. doi:10.3390/s20216300

[24] Garcia, K. D., de Sá, C. R., Poel, M., Carvalho, T., Mendes-Moreira, J., Cardoso, J. M., & Kok, J. N. (2021). An ensemble of autonomous auto-encoders for human activity recognition. Neurocomputing, 439, 271-280. https://doi.org/10.1016/j.neucom.2020.01.125

[25] Wu, C., Wang, Z., Hu, S., Lepine, J., Na, X., Ainalis, D., & Stettler, M. (2020). An automated machine-learning approach for road pothole detection using smartphone sensor data. Sensors, 20(19), 5564. DOI: 10.3390/s20195564

[26] Hasanuzzaman, M. D., Kabir, M. A., & Ahmed, M. T. (2021). Transpiration effect on unsteady natural convection boundary layer flow around a vertical slender body. Results in Engineering, 12, 100293.doi: https://doi.org/10.1016/j.rineng.2021.100293

[27] Ramanujam, E., Perumal, T., & Padmavathi, S. (2021). Human activity recognition with smartphone and wearable sensors using deep learning techniques: A review. IEEE Sensors Journal, 21(12), 13029-13040. DOI 10.1109/JSEN.2021.3069927

[28] Rana, S. (2021). Vibration based pavement roughness monitoring system using vehicle dynamics and smartphone with estimated vehicle parameters. Results in Engineering, 12, 100294. https://doi.org/10.1016/j.rineng.2021.100294

[29] Kołakowska, A., Szwoch, W., & Szwoch, M. (2020). A review of emotion recognition methods based on data acquired via smartphone sensors. Sensors, 20(21), 6367.doi:10.3390/s20216367

[30] Nooruddin, S., Islam, M. M., Sharna, F. A., Alhetari, H., & Kabir, M. N. (2022). Sensor-based fall detection systems: a review. Journal of Ambient Intelligence and Humanized Computing, 1-17.DOI: https://doi.org/10.1007/s12652-021-03248-z

[31] Straczkiewicz, M., James, P., & Onnela, J. P. (2021). A systematic review of smartphone-based human activity recognition methods for health research. NPJ Digital Medicine, 4(1), 148. https://doi.org/10.1038/s41746-021-00514-4

[32] Pias, T. S., Eisenberg, D., & Fresneda Fernandez, J. (2022). Accuracy improvement of vehicle recognition by using smart device sensors. Sensors, 22(12), 4397. https://doi.org/10.3390/s22124397

[33] Cerone, G. L., Botter, A., & Gazzoni, M. (2019). A modular, smart, and wearable system for high density sEMG detection. IEEE Transactions on Biomedical Engineering, 66(12), 3371-3380. DOI: 10.1109/TBME.2019.2904398

[34] Balasubramanian, V., Otoum, S., & Reisslein, M. (2022). VeNet: hybrid stacked autoencoder learning for cooperative edge intelligence in IoV. IEEE Transactions on Intelligent Transportation Systems, 23(9), 16643-16653. DOI: 10.1109/TITS.2022.3170372

[35] Vital, T. P. R., Nayak, J., Naik, B., & Jayaram, D. (2021). Probabilistic neural network-based model for identification of Parkinson's disease by using voice profile and personal data. Arabian Journal for Science and Engineering, 46(4), 3383-3407. DOI: https://doi.org/10.1007/s13369-020-05080-7

[36] Vital, T. P., Sangeeta, K., & Kumar, K. K. (2021). Student classification based on cognitive abilities and predicting learning performances using machine learning models. International Journal of Computing and Digital Systems, 10(1), 63-75. DOI: http://dx.doi.org/10.12785/ijcds/100107

[37] Terlapu, P. V., Gedela, S. B., Gangu, V. K., & Pemula, R. (2022). Intelligent diagnosis system of hepatitis C virus: A probabilistic neural network based approach. International Journal of Imaging Systems and Technology, 32(6), 2107-2136. DOI: https://doi.org/10.1002/ima.22746

[38] Balasankar, V., Penumatsa, S. V., & Terlapu, P. R. V. (2020). Intelligent socio-economic status prediction system using machine learning models on Rajahmundry AP, SES dataset. Indian Journal of Science and Technology, 13(37), 3820-3842. DOI : https://doi.org/ 10.17485/IJST/v13i37.1435

[39] Peppas, K., Tsolakis, A. C., Krinidis, S., & Tzovaras, D. (2020). Real-time physical activity recognition on smart mobile devices using convolutional neural networks. Applied Sciences, 10(23), 8482. doi:10.3390/app10238482

[40] Thakur, D., Biswas, S., Ho, E. S., & Chattopadhyay, S. (2022). Convae-lstm: Convolutional autoencoder long short-term memory network for smartphone-based human activity recognition. IEEE Access, 10, 4137-4156. DOI:10.1109/ACCESS.2022.3140373

[41] Khan, Y. A., Imaduddin, S., Singh, Y. P., Wajid, M., Usman, M., & Abbas, M. (2023). Artificial Intelligence Based Approach for Classification of Human Activities Using MEMS Sensors Data. Sensors, 23(3), 1275. DOI: https://doi.org/10.3390/s23031275

[42] Straczkiewicz, M., James, P., & Onnela, J. P. (2019). A systematic review of smartphone-based human activity recognition for health research. arXiv preprint arXiv:1910.03970.

[43] Ignatov, A. (2018). Real-time human activity recognition from accelerometer data using Convolutional Neural Networks. Applied Soft Computing, 62, 915-922. http://dx.doi.org/doi:10.1016/j.asoc.2017.09.027

[44] Mutegeki, R., & Han, D. S. (2020, February). A CNN-LSTM approach to human activity recognition. In 2020 international conference on artificial intelligence in information and communication (ICAIIC) (pp. 362-366). IEEE. DOI: 10.1109/ICAIIC48513.2020.9065078

[45] Huang, W., Zhang, L., Gao, W., Min, F., & He, J. (2021). Shallow convolutional neural networks for human activity recognition using wearable sensors. IEEE Transactions on Instrumentation and Measurement, 70, 1-11. DOI 10.4108/icst.mobicase.2014.257786

[46] Rahman, M. A., Mia, Y., Masum, M. R., Abid, D. M. H., & Islam, T. (2022, June). Real Time Human Activity Recognition from Accelerometer Data using Convolutional Neural Networks. In 2022 7th International Conference on Communication and Electronics Systems (ICCES) (pp. 1394-1397). IEEE. DOI: 10.1109/ICCES54183.2022.9835797

[47] de Aquino e Aquino, G., Serrão, M. K., Costa, M. G. F., & Costa-Filho, C. F. F. (2022, April). Human Activity Recognition from Accelerometer Data with Convolutional Neural Networks. In XXVII Brazilian Congress on Biomedical Engineering: Proceedings of CBEB 2020, October 26–30, 2020, Vitória, Brazil (pp. 1603-1610). Cham: Springer International Publishing. DOI: 10.1007/s41050-021-00028-8

[48] Aquino, G., Costa, M. G., & Costa Filho, C. F. (2022). Explaining One-Dimensional Convolutional Models in Human Activity

Recognition and Biometric Identification Tasks. Sensors, 22(15), 5644. https://doi.org/10.3390/s22155644

[49] Gamble, J. A., & Huang, J. (2020, August). Convolutional neural network for human activity recognition and identification. In 2020 IEEE International Systems Conference (SysCon) (pp. 1-7). IEEE. DOI: 10.1109/SysCon47679.2020.9275924

[50] Aquino, G., Costa, M. G. F., & Filho, C. F. F. C. (2023). Explaining and Visualizing Embeddings of One-Dimensional Convolutional Models in Human Activity Recognition Tasks. Sensors, 23(9), 4409. DOI: 10.3390/s23094409