Egyptian Informatics Journal 24 (2023) 100391

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

Full length article

Diagnosis of patellofemoral osteoarthritis using enhanced sequential deep learning techniques

Mai Ramadan Ibraheem [a], Saleh Naif Almuayqil [b], A. A. Abd El-Aziz [b,c], Medhat A. Tawfeek [d,e,*], Fatma M. Talaat [f]

[a] Dept. Information Technology, Faculty of Computers and Information, Kafrelsheikh University, Egypt
[b] Dept. of Information Systems, College of Computer and Information Sciences, Jouf University, Saudi Arabia (KSA)
[c] Dept. of Information Systems and Technology, Faculty of Graduate Studies for Statistical Research, Cairo University, Egypt
[d] Dept. of Computer Science, College of Computer and Information Sciences, Jouf University, Saudi Arabia (KSA)
[e] Dept. of Computer Science, Faculty of Computers and Information, Menoufia University, Egypt
[f] Dept. Machine Learning & Information Retrieval Department, Faculty of Artificial Intelligence, Kafrelsheikh University, Egypt

ARTICLE INFO

Keywords:
SEMG Signals
Data Augmentation
Time Series Analysis
Lower Limbs
Sequence Data

ABSTRACT

The surface electromyography (sEMG) signal is a complex interference pattern resulting from the electrical activity of contracting muscles, which directly correlates with muscle activity and exercise level. Patellofemoral osteoarthritis (PF OA) refers to the softening and destruction of the articular cartilage of the knee cap, which is important to diagnose in the early phase. The aim of this study is to develop a predictive model, called the Enhanced Sequencing Deep Learning (PESDL) Algorithm, to detect PF OA from the sEMG signal. The algorithm consists of four main modules: (i) Data Acquisition Module (DAM), (ii) Signal Preprocessing Module (SPM), (iii) Data Augmentation and Concatenation Module (DACM), and (iv) Lower Limb Classification (LLC). The sEMG signals from five core muscles of the lower limb were acquired during stepping stairs up and down, and machine learning was used to obtain the muscle activation data signals. The acquired data was preprocessed, augmented, concatenated, and shuffled. Three feedforward deep networks (RNN, LSTM, and GRU) were used to classify the lower limb sEMG time-sequence data, with the GRU network showing better performance than the other two. The model's performance was evaluated using seven performance measures and 10-fold cross-validation. The maximum values of feedforward deep networks (RNN), LSTM, and GRU for the five muscles (RF, BF, VM, ST, and FX) using seven performance measures were as follows: RNN (0.961, 0.154, 0.943, 0.244, 1.0, 1.0, 0.899), LSTM (0.967, 0.147, 0.945, 0.217, 0.997, 1.0, 0.927), and GRU (0.976, 0.111, 0.949, 0.212, 0.996, 0.998, 0.938), respectively, listed in the order of accuracy, loss, validation accuracy, validation loss, recall, precision, and F1-score. Comparisons with other state-of-the-art models using the same datasets demonstrated the effective performance of the predictive model. The results suggest that RNN models, particularly GRU and LSTM, can be effective for detecting PF OA, and the specific RNN architecture used can have a significant impact on performance.

1. Introduction

This section discusses some important issues, such as Patellofemoral osteoarthritis (PF OA), Long-Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Recurrent Neural Networks (RNN), and the primary contribution of this study.

1.1. Patellofemoral osteoarthritis (PF OA)

Patellofemoral osteoarthritis (PF OA) is the softening and damage of the articular cartilage of the knee cap, which is the cartilage lining beneath the knee cap [1]. Progressive rehabilitation activities are necessary for PF OA sufferers to regain full muscle function, as instability or reduced core strength may contribute to the increased risk of lower extremity injuries [2]. Since the stability of the trunk and pelvis is essential for limb movement, surface electromyographies (sEMG) are

* Corresponding author.
E-mail address: Maelaarg@ju.edu.sa (M.A. Tawfeek).

https://doi.org/10.1016/j.eij.2023.100391
Received 18 January 2023; Received in revised form 13 April 2023; Accepted 31 July 2023
Available online 10 August 2023

signals used to track the electrical function of muscle contraction and the manifestation of muscle exhaustion. These signals play a significant role in the assessment; however, they contain a lot of noise, resulting in a significant error margin in the recorded data [1]. Biological intelligence processes data in little chunks, built on previous data and modified as new data comes in. sEMG data have data points arranged in a sequence or temporal series that must be treated as a full sequence to the network at once [3].

1.2. Recurrent neural networks (RNN)

Recurrent neural networks (RNNs) are networks that have feedback connections between neurons and are capable of handling sequential data. In RNNs, the output of the previous step is used as input for the current step. RNNs have a memory that keeps track of all the information that has been processed. Since the network performs the same operation on all inputs or hidden layers to produce an output, it uses the same parameters for each input, which simplifies the model's parameters compared to other neural networks. RNNs use a hidden layer to address this issue. The hidden state, which stores a portion of a sequence's information, is the most significant and essential feature of RNNs [4]. Fig. 1 illustrates an RNN. The general form of the network can be expressed by Equation (1), where $X \in \mathbb{R}^{t\times d}$ represents the input data with t steps and d channels. The output $X \in \mathbb{R}^{t\times n_c}$ has dimensionality n_c and the same length, t. Θ denotes the parameters of the network and $\mathscr{L}$ is a one-to-one correspondence in which each time step corresponds to an output time steps [3].

$$Y = \mathscr{L}_{\Theta}(X) \tag{1}$$

A fundamental limitation of RNNs is their difficulty in processing long sequences due to limited access to input information [3]. RNNs have access to less input information; they only consider the first few features rather than entire sequences. As a result, RNN isn't as good as other recurrent layers with more complex layers at digesting long sequences [5]. In section 3, we will discuss the overall steps in RNN to enhance its performance.

1.3. Long Short-Term memory (LSTM)

LSTM solves the problem of vanishing gradients by adding the ability to transfer data across many timesteps. Input data sequences can be loaded at any time, carried to a later timestep, and retrieved when needed. The information is preserved for later use, and older signals are also prevented from being lost within the process [6]. Learning long-term relationships in RNN has been found as a limitation issue. LSTM consecutively addresses this issue by enriching the hidden node parameters to overcome looping problems and customize the acquiring and releasing of states based on the input sequence [6]. Thus, the states can be activated based on the short-term episode while holding the network states active, providing the result of long-term memory. RNN is not as good as LSTM at learning the sequences [7]. LSTM has four weight/bias pairs rather than RNN, which has a bias and a single weight. The four weight/bias pairs are as follows [8]: (i) Forget Gate Layer, (ii) Input Gate Layer, (iii) Output Gate Layer, (iv) State Gate Layer.

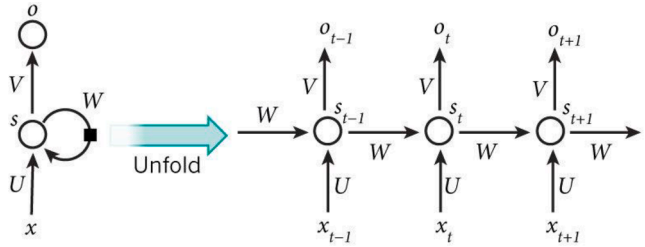


Fig. 1. RNN basic flow.

Equation 2 provides a description of the forward pass LSTM cell equations. The forget and input gates $f_t(2b)$ and $i_t(2a)$ respectively, control the previously hidden state h_{t-1} and the current input x_t contribution to the cell state c_t [9]. Fig. 2 depicts the internal structure of LSTM and shows that the *tanh* hyperbolic tangent function state is used to filter the output of the hidden state as well as scale the activation of the input, output, and forgetting gates [10].

$$i_t = \sigma(W_i \bullet [h_{t-1}, x_t] + b_i) \tag{2a}$$

$$f_t = \sigma(W_f \bullet [h_{t-1}, x_t] + b_f) \tag{2b}$$

$$o_t = \sigma(W_o \bullet [h_{t-1}, x_t] + b_o) \tag{2c}$$

$$\bar{c}_t = \tanh(W_c \bullet [h_{t-1}, x_t] + b_c) \tag{2d}$$

$$c_t = f_t \bullet c_{t-1} + i_t \bullet \bar{c}_t \tag{2e}$$

$$h_t = o_t \bullet \tanh(c_t) \tag{2f}$$

The activation of LSTM cells given by Equation 2 shows the requirement for previous knowledge in time. Thus, the network parameter optimization via stochastic gradient descent is performed through the time steps in the order of the input data. Correspondingly, the data input is split into smaller windows to decrease the time depth of BP backpropagation. Typically, in an FFNN structure, the number of variables and time duration of the input layer are fixed. There are several hidden layers in fully connected LSTM layers, and an output layer serves as a classification softmax transfer function. The input and output layers' time lengths are always the same for performing a sequence-to-sequence classification configuration. The LSTM cells' states consequently lasted for consecutive training sequences [11].

1.4. Gated Recurrent Unit (GRU)

Although GRU may not have as much representational power as LSTM, it operates on the same premises and is more streamlined, resulting in lower operating costs. To address the issue with ordinary RNN, GRU integrates two gate operating mechanisms called the update gate and the reset gate [12]. The update gate is responsible for figuring out the value of prior data that is required for the next cell. It is powerful for the model to copy all the previous data to minimize the risk of vanishing gradients. The reset gate is responsible for deciding how much of the previous information can be discarded [13].

The update gate shown by Eq. (3a) has a single weight to customize the influence of the prior hidden state h_{t-1} on the updated. Contrary to the reset gate given by Eq. (3b), which determines the number of past states that are discarded, the memory cells updates $\bar{h}_t$ are shown by Eq. (3c). The output is a computed using the updated h_t state and the prior $h_{(t-1)}$ state, as determined by Eq. (3d). With fewer parameters, GRU almost has the advantages of LSTMs over RNNs [14]. Therefore, GRU

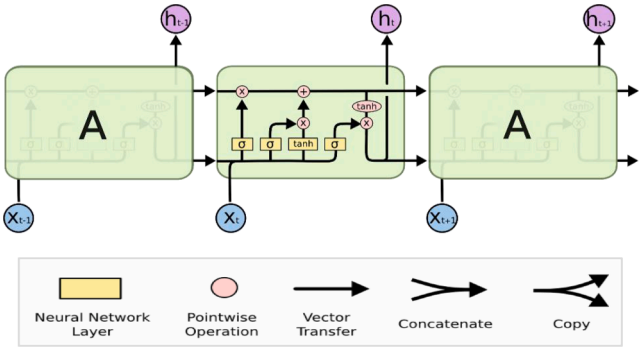


Fig. 2. Internal structure of LSTM.

networks lead to faster training and inference.

$$z_t = \sigma(W_i \bullet [h_{t-1}, x_t]) \tag{3a}$$

$$r_t = \sigma(W_i \bullet [h_{t-1}, x_t]) \tag{3b}$$

$$\overline{h}_t = \tanh(W_i \bullet [r_t{*}h_{t-1}, x_t]) \tag{3c}$$

$$h_t = (1 - z_t){*}h_{t-1} + z_t{*}\overline{h}_t \tag{3d}$$

Relevant information is first activated and stored by the reset gate using the previous time step. Then the corresponding weights are multiplied by the input vector and hidden state. The reset gate and the previously hidden state are then wisely multiplied. The resulting sequence is generated by summing up the previously stated elements using the non-linear activation function. The update z_t, reset r_t, previously hidden activation h_{t-1}, and memory state update $\overline{h}_t$ are shown in the GRU internal structure given in Fig. 3 [13]. According to Equation 3, the updated state ht and the prior state are averaged together to get the final state $h_{(t-1)}$.

The internal structure of GRU shows that It only has two gates, without the output gate and internal memory as in LSTMs with three gates. In an LSTM, an update gate connects the input and target gates, but GRU reset gates are directly connected to the previous hidden state [13]. Through the input and target gates, the reset gate in LSTM can be obtained.

1.5. Contribution

The main contribution of this paper is to develop a predictive model to investigate and interpret PF OA from the sEMG signal. The proposed Enhanced Sequencing Deep Learning (PESDL) algorithm contains four main modules, which are; (i) Data Acquisition Module (DAM), (ii) Signal Preprocessing Module (SPM), (iii) Data Augmentation, and Concatenation Module (DACM)., and (iv) Lower Limb Classification (LLC). It aims to explore the ability of different types of feedforward networks to learn and memorize the sequence data in terms of sEMG. It utilizes RNN, LSTM, and GRU in representing and classifying sEMG sequential data signals.

The proposed PESDL was trained using the EMG dataset in the lower limb, which contains 66 records for knee abnormality along with normal ones, with five attributes describing the corresponding muscles measured. The dataset was augmented using a time-series generator that yielded samples and their targets. The augmentation step resulted in 1056 records of EMG sequence signals for the five corresponding muscles.

The remainder of the paper is organized as follows: Section 2 discusses related work. Section 3 presents the proposed framework phases, and section 4 shows figures, the used datasets, and the experimental results. Finally, the discussion and conclusion are explained in Section 5.

2. Literature review

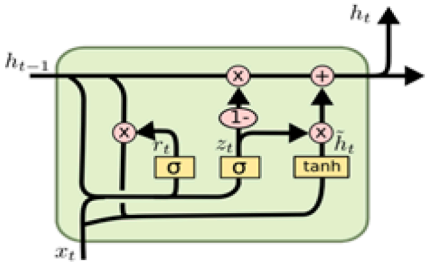
Several attempts have been made to address the difficulties mentioned above using deep learning architectures.

For example, Ibraheem et al. [15] applied three time–frequency representation techniques, namely scalogram, spectrogram, and persistence spectrum, to the 1D sEMG signal of knee muscles. The objective was to detect knee abnormality by straightening the knee and obtaining 2D projected images that were input into a convolutional neural network (CNN) model. The experiments were conducted using 10-fold cross-validation, with the number of kernels increasing as the model layers grew. The fully connected layers were adjusted based on the loss value, and the hyper-parameters of the dropout parameters and ReLU activation function were tuned for optimal performance. The researchers used the same public EMG dataset in Lower Limb as in their manuscript. Their findings indicate that the scalogram image representation outperformed the spectrogram and persistence spectrum in detecting knee abnormality, with a maximum accuracy of 84%.

Bansal et al. [16] aimed to investigate significant differences in the movements associated with knee muscles, gait, leg extension from a seated position, and leg flexion upwards in regular and abnormal sEMG data. To achieve this, they employed a range of machine learning techniques such as iforest, KNN, and boosting algorithms to enhance classifier accuracy and predict the three distinct movements for normal and abnormal sets of data. The researchers started by denoising and filtering the sEMG data. They used the same public EMG dataset in Lower Limb that was utilized in their manuscript, achieving an accuracy rate of 85% without implementing anomaly detection techniques, as was also done in our model.

Zhu et al. [17] investigated the effect of sEMG preprocessing and prediction on walking data collected from 10 adults (five males and five females). They performed preprocessing analysis using bandpass/principal components and independent components. Then they imported the processed data into support vector regression for training and prediction. A machine learning support vector regression is based on statistical learning, and it adopts generalized learning by minimizing the risk. They reached 94.54% ± 2.98 prediction accuracy of knee motion with various drawbacks in the types of movements studied. Moreover, due to the complexity of sEMG movement signals, choosing the best place for signal capture and studying the inherent characteristics of sEMG from a time perspective are significantly needed.

Wang et al. [18] proposed a surface electromyography (sEMG)-based real-time stable control gait switching approach for the lower leg that achieves intention recognition and muscle fatigue. The stability of the human exoskeleton system is tested through different gaits. They considered the relationship between the hip, knee, and ankle joints. They defined the values for detecting the crossing obstacle as −1, the reset of the initial state of P0 (stop), and the value 1 for the state P1 (walk). They implemented recognition using an LSTM deep learning model best suited to motion detection based on studying the sEMG and intrinsic joint properties from a temporal perspective, utilising LSTM memory cells that can access and keep data for long periods of time. Their model achieved an accuracy of nearly 99%, with the drawback that the model was only experimented on and trained on healthy people. Additionally, patient sEMG data may not be comprehensive and should be thought of as adapting to various walking environments.



$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$$
$$\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Fig. 3. Internal structure of GRU.

Li et al. [19] researched the relationship between joint movement and surface electromyography (sEMG), creating a valid reference for exoskeleton control. During normal adult walking, the sEMG and the associated movement of the knee were measured. A random forest with principal component analysis (RFPCA) was used to establish the model for knee movement in sEMG.

The estimated model RFPCA was compared to a back propagation neural network with principal component analysis to validate their findings (BPPCA). RFPCA was found appropriate for all experiment participants and takes less time to execute. RFPCA had a mean square error of about 5, which was lower than BPPCA, which had errors ranging from 7 to 25. The researchers concluded that the RFPCA approach for estimating knee movement from sEMG is practical and might be employed for exoskeleton control and motion analysis.

Tsinganos et al. [20] used two datasets to evaluate existing data augmentation methodologies for electromyography signals (i.e., magnitude warping, overlapping windows, wavelet decomposition, synthetic sEMG models, and additive noise). The AtzoriNet, a modified version of a lightweight CNN for gesture recognition, was used in the test. A collection of metrics and visualizations (such as classification accuracy, the davies–Bouldin index, and the silhouette score) aid in evaluating and offering information on their performance. Over the two benchmark datasets, signal magnitude warping and wavelet decomposition significantly boost classification accuracy (up to 16%).

Morbidoni et al. [21] proposed a deep learning approach for sEMG-based classification and prediction of the foot-floor contact signal in different natural walking conditions. In [21], 23 healthy individuals were asked to record sEMG signals from eight lower-limb muscles while walking on flat ground, following an eight-shaped pattern that included natural acceleration, a deceleration curve, and reversing. They concluded that utilizing a multilayer perceptron to learn hidden features gave acceptable performance while avoiding feature engineering through a comprehensive evaluation. The results showed an average categorization accuracy of 94.9 for subjects who had received training and 93.4 for untrained people, with heel-strike and toe-strike having mean absolute differences of 21.6 ms and 38.1 ms, respectively, between phase transition timing forecasts and footswitch data.

Khowailed and Abotabl [22] introduced the neural muscular activation detection (NMAD) framework, which uses deep learning to identify muscle activation. Not only does the proposed method greatly enhance the time detection accuracy, but it can also adapt to varying levels of interference and signal-to-noise ratio due to its training nature. The key notion is that, rather than relying exclusively on heuristic aspects of the sEMG signal to predict activation timing, an artificial neural network can be trained to do so. Because the sEMG signal is time series, a recurrent neural network (RNN) is a good candidate for determining the timing of muscle activity. The findings suggest that a properly trained RNN can outperform state-of-the-art algorithms in terms of accuracy. Furthermore, the trained neural network can work reliably in a wide range of SNR and cross-talk conditions.

In evaluating sEMG quality, Akhundov et al. [23] assessed the performance of supervised and unsupervised artificial neural networks (ANNs). The ANNs were trained (n = 28,000), tested (n = 12,000), and evaluated (n = 47,000) in classifying signals into four categories. When compared to supervised ANNs, unsupervised ANNs showed a 30–40% improvement in classification accuracy (>98%). AlexNet had the best accuracy (99.55%) and the fewest false classifications.

For studying the continuous estimation of sEMG of knee joint angle, Liu et al. [24] suggested predicting knee angle using a convolutional neural network (CNN) model. Normal walking studies were conducted with a six-channel sEMG acquisition device and an optical motion capture system to determine the actual and intended knee angle. The suggested model includes two other neural network models: the convolutional neural network (CNN) and the backpropagation neural network (BPNN). Their results reached the BPNN and the CNN models, which can predict knee angles with greater accuracy.

The proposed approach extends the previous work by using time series representation and classification techniques to enhance the predicted results. Further, the proposed model performs preprocessing, augmentation, and shuffling for better and balanced training data. Different fine-tuning settings have also been set to increase classification performance for knee abnormalities. The proposed novel deep classification model for sEMG sequential data has reported significant results.

The previous algorithms used for detecting PF OA using sEMG signals have some limitations includeing:

1. Limited feature extraction and selection methods: Previous algorithms used basic time and frequency domain features, such as mean, variance, and power spectral density, which may not capture the full complexity of sEMG signals.
2. Limited data augmentation techniques: Previous algorithms used simple data augmentation techniques, such as random noise and time-shifting, which may not effectively enhance the diversity and variability of the sEMG data.
3. Limited deep learning architectures: Previous algorithms used only a few deep learning architectures, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory (LSTM) networks, which may not fully exploit the temporal dependencies and sequential nature of sEMG signals.

The proposed PESDL algorithm addresses these limitations by incorporating novel features, such as higher-order statistics and wavelet transforms, advanced data augmentation techniques, such as time-series data generation and concatenation, and a combination of deep learning architectures, including RNNs, LSTMs, and gated recurrent units (GRUs).

The contributions of the proposed algorithm can be summarized as follows:

1. Enhanced feature extraction and selection: The PESDL algorithm incorporates novel higher-order statistical and wavelet-based features that capture the non-Gaussian and non-stationary nature of sEMG signals more accurately than previous algorithms.
2. Improved data augmentation techniques: The PESDL algorithm uses more advanced time-series data generation and concatenation techniques that enhance the diversity and variability of the sEMG data more effectively than previous algorithms.
3. A combination of advanced deep learning architectures: The PESDL algorithm combines three different deep learning architectures (RNNs, LSTMs, and GRUs) to fully exploit the temporal dependencies and sequential nature of sEMG signals and achieve higher classification accuracy than previous algorithms.

Overall, the proposed PESDL algorithm addresses some of the limitations of previous algorithms used for detecting PF OA using sEMG signals and contributes to the research gap by providing a more accurate and reliable method for early detection of PF OA, which could help to improve clinical outcomes and patient quality of life.

The proposed Enhanced Sequencing Deep Learning (PESDL) Algorithm contains four main modules, including the Lower Limb Classification (LLC) module, which is responsible for improving the performance of deep neural networks. Specifically, the LLC module utilizes three types of recurrent neural networks (RNN), including RNN, LSTM, and GRU, to classify the lower limb sEMG time-sequence data. The GRU network, in particular, showed better performance than the other two in detecting PF OA. Additionally, we used a time-series generator to augment the acquired data, resulting in more samples and targets for the classification task. We also shuffled the data to prevent overfitting. Overall, the combination of these techniques in the LLC module allowed for improved performance of the deep neural networks in detecting PF OA from sEMG signals.

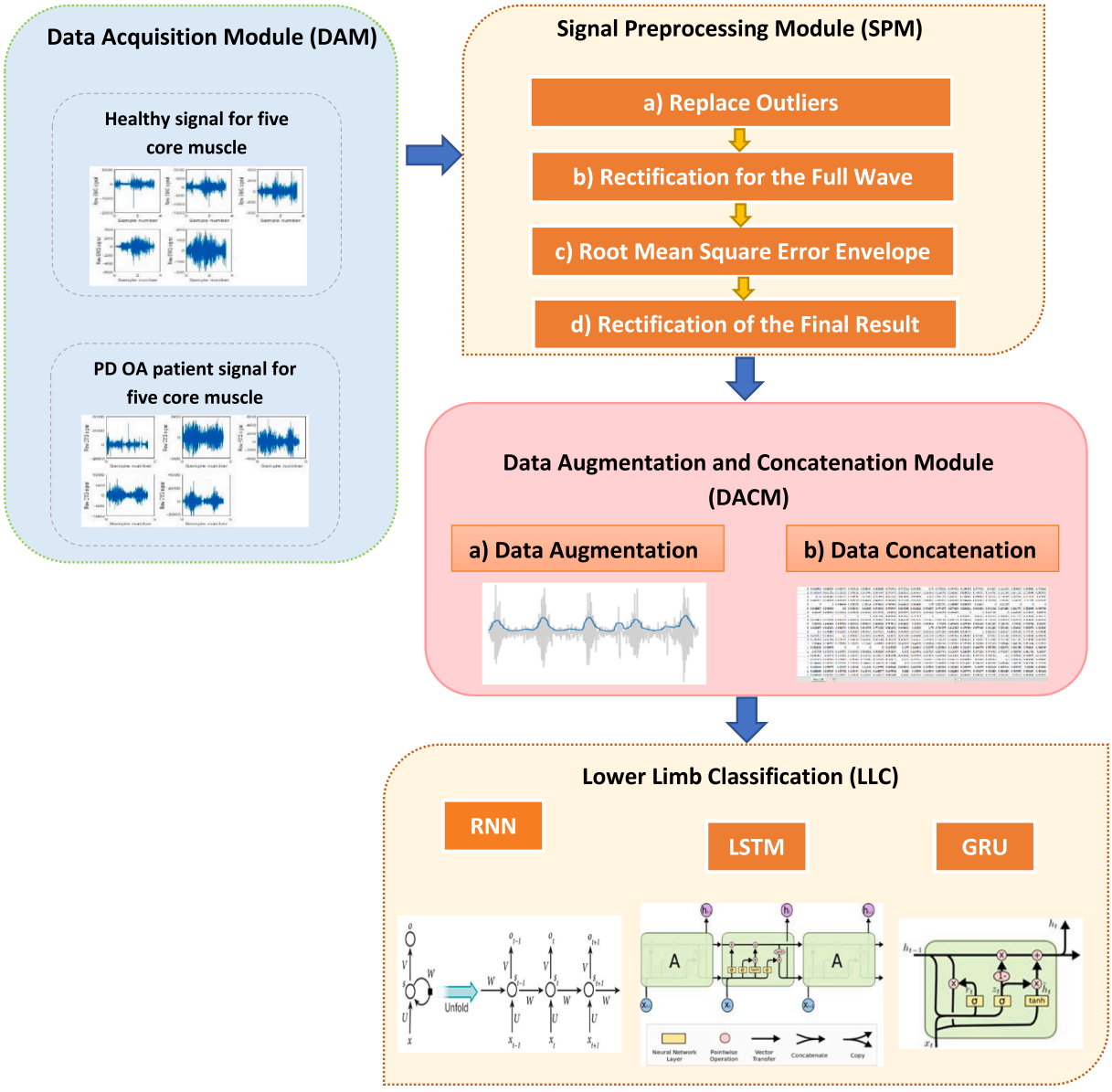


Fig. 4. Proposed Enhanced Sequencing Deep Learning (PESDL).

3. Proposed Enhanced Sequencing Deep Learning (PESDL)

The following is the main contribution of this paper. The proposed Enhanced Sequencing Deep Learning (PESDL) Algorithm contains four main modules, as shown in Fig. 4, which are; (i) Data Acquisition Module (DAM), (ii) Signal Preprocessing Module (SPM), (iii) Data Augmentation and Concatenation Module (DACM)., and (iv) Lower Limb Classification (LLC).

In DAM, the sEMG signals were acquired from five core muscles of the lower limb in about 200 reads from healthy and patient adults during stepping up and down stairs. The five core muscles, TrA, VMO, GM, VL, and ML, were acquired to obtain the muscle activation data signal. The acquired data is then preprocessed, augmented, concatenated, and shuffled. The augmentation step was performed using a time-series generator yielding samples and their targets, resulting in 1056 records for the five muscles of interest. Three feedforward deep networks are selected to represent and classify the lower limb sEMG time-sequence data. RNN, LSTM, and GRU are used in the classification of the time-sequence data because they can control the input flow and prevent the negative effects from achieving a timely and accurate interpretation of PF OA.

3.1. Data Acquisition Module (DAM)

This database contains 22 male samples, 11 with knee abnormalities and 11 with healthy people. We used electromyography and goniometry equipment from MWX8 Datalog Biometrics to acquire these data. We used a goniometer for the knee and four electrodes (Vastus Medialis, Semitendinosus, Biceps Femoris, and Rectus Femoris). Fig. 5 shows samples of the acquired data for one of the five muscles of interest RF in normal people. Fig. 6 shows the samples for muscle FX in abnormal people.

3.2. Signal preprocessing Module (SPM)

The quality of the sEMG signal captured is affected by the quantity, depth, tissue content, and location of the activated fibres in the muscle. These variables can mask undesirable signals or cancel out the signal

	0	1	2	3	...	36896	36897	36898	36899
0	0.590476	0.352480	0.326590	0.292428	...	1.0	1.0	1.0	0.0
1	0.697619	0.569191	0.543353	0.587467	...	1.0	1.0	1.0	0.0
2	0.642857	0.569191	0.500000	0.509138	...	1.0	1.0	1.0	0.0
3	0.661905	0.608355	0.566474	0.569191	...	1.0	1.0	1.0	0.0
4	0.428571	0.195822	0.260116	0.686684	...	1.0	1.0	1.0	0.0
5	0.642857	0.530026	0.500000	0.490862	...	1.0	1.0	1.0	0.0
6	0.000000	0.000000	0.066474	0.255875	...	1.0	1.0	1.0	0.0
7	0.411905	0.313316	0.306358	0.469974	...	1.0	1.0	1.0	0.0
8	0.626190	0.548303	0.523121	0.548303	...	1.0	1.0	1.0	0.0
9	0.733333	0.665796	0.523121	0.509138	...	1.0	1.0	1.0	0.0
10	0.411905	0.156658	0.023121	0.391645	...	0.0	0.0	0.0	0.0
11	0.642857	0.587467	0.500000	0.530026	...	1.0	1.0	1.0	0.0
12	0.642857	0.548303	0.566474	0.569191	...	1.0	1.0	1.0	0.0
13	0.590476	0.509138	0.500000	0.509138	...	1.0	1.0	1.0	0.0
14	1.000000	1.000000	1.000000	1.000000	...	1.0	1.0	1.0	0.0
15	0.642857	0.548303	0.500000	0.548303	...	1.0	1.0	1.0	0.0
16	0.626190	0.548303	0.479769	0.509138	...	1.0	1.0	1.0	0.0
17	0.697619	0.490862	0.436416	0.530026	...	1.0	1.0	1.0	0.0
18	0.642857	0.548303	0.566474	0.569191	...	1.0	1.0	1.0	0.0
19	0.376190	0.195822	0.000000	0.000000	...	1.0	1.0	1.0	0.0
20	0.750000	0.686684	0.566474	0.569191	...	1.0	1.0	1.0	0.0
21	0.840476	0.647520	0.609827	0.469974	...	1.0	1.0	1.0	0.0
22	0.626190	0.608355	0.543353	0.569191	...	1.0	1.0	1.0	0.0
23	0.642857	0.587467	0.523121	0.490862	...	1.0	1.0	1.0	0.0
24	0.626190	0.490862	0.479769	0.409922	...	1.0	1.0	1.0	0.0
25	0.626190	0.548303	0.500000	0.509138	...	1.0	1.0	1.0	0.0
26	0.733333	0.665796	0.523121	0.509138	...	1.0	1.0	1.0	0.0
27	0.642857	0.587467	0.566474	0.569191	...	1.0	1.0	1.0	0.0
28	0.642857	0.530026	0.479769	0.509138	...	1.0	1.0	1.0	0.0
29	0.500000	0.430809	0.413295	0.430809	...	1.0	1.0	1.0	0.0
30	0.678571	0.569191	0.543353	0.509138	...	1.0	1.0	1.0	0.0
31	0.230952	0.255875	0.109827	0.313316	...	1.0	1.0	1.0	0.0
32	0.626190	0.530026	0.523121	0.509138	...	1.0	1.0	1.0	0.0

[33 rows x 36900 columns]

Fig. 5. Normal samples for RF muscle.

from the muscle we're trying to record. The absolute value of the signal is used to apply full-wave rectification. Rectification is the process of transforming a negative swing into a positive swing. Fig. 7 shows the raw samples of the data before preprocessing.

Rectification is accomplished by smoothing the signal by high-passing it. High-pass filtering effectively eliminates the effects of maintaining a constant force or of repeated fluctuations in the strength of the sEMG signals [25]. A high-pass filter works effectively to remove the oscillation produced by overlaid artfacts. By using a design function of order 4, a pass band frequency of 75 kHz, and a sampling frequency of 1000 Hz, this step builds a high pass filter [26]. The developed high-pass filter efficiently converts signals in the time domain with variable resolution. The algorithm for the preprocessing step is given in Algorithm 1. The output of the rectification step is shown in Fig. 8.

3.3. Data augmentation and Concatenation Module (DACM)

This module combines two submodules; (i) Data Augmentation and (ii) Data Concatenation and Shuffle.

3.3.1. Data augmentation

Data augmentation is a set of methods for increasing the amount and variety of data available for training models for a certain job without acquiring additional data. This group of strategies helps the model learn the extent of intraclass variation that can be seen by changing existing labelled samples into synthetic or artificial data [27]. Generating additional data while keeping the correct label is one of the challenging parts of data augmentation. Numerous tasks require domain expertise, which may be challenging to acquire in the case of sEMG due to significant within-subject variability. In other words, the same person can execute the same gesture in different ways, and the quality of the recorded signal is influenced by things like fatigue and sweat [28]). Because enhanced data are supposed to reflect a larger set of possible data points, data augmentation is one of the techniques for dealing with the problem of overfitting. It reduces the difference in the training, validation, and testing sets [29].

Augmentation approaches for biosignal processing have been created as a series of simple alterations in sequence (e.g., shuffling, warping, additive noise, or time). In addition, the authors have devised alternative approaches for generating augmented data based on sub-optimally matched sequences [30]. Compared to previous approaches, their methodology performed better or equivalently across various datasets, including ECG and EEG signals. Several sEMG-based gesture recognition algorithms have been tested using a set of domain-specific augmen,tations. Sensor noise simulation, electrode displacement simulation, and muscle fatigue approximation are among them [20].

This study generated enhanced data using a sliding window technique and non-overlapping signal segments. The augmentation study reveals that using a sliding window technique improves performance significantly [31]. Data augmentation is used to create a newly labelled signal x^*i, y^*i from an existing one x_i, y_i without changing the class label, i.e., $y\_i = y^*i$. Data augmentation is achieved in this study by smoothing the signal of sEMG sequences using the Root-Mean-Squared method (RMS) [31]. Convolution is also used to improve the looping of the window movement, as shown in Fig. 9.

In addition to considering the signal's time- and frequency-domain properties, this method also eliminates boundary difficulties by keeping the windows narrow. Moving average smoothing (moving window)

Signal Preprocessing Algorithm (SPA)

Input:

- The five considered muscles' raw sEMG signal.

Output:

- Processed sEMG signal

Steps:

1: **Raw sEMG signals for each of the five considered muscles**
2: take the absolute value of the signal to rectify the sEMG signal's whole wave
3: apply an order four low pass filter with a 10 Hz cutoff frequency.
4: Using an order 4 design function, construct a high pass filter, with a sampling rate of 1000 HZ, and a passband frequency of 75 kHz.
5: Rectify the final result
8: **End**

corresponds to the signal power (energy), which could be used to estimate muscular exertion. The quantity for a certain time interval is defined by a sliding average. It's the best option for smoothing with a time of 50 to 100 ms or less. The augmentation step resulted in 1056 records of sEMG sequence signals for each of the five muscles measured. The algorithm for the augmentation step is given by Algorithm 2.

3.3.2. Data Concatenation and shuffle

This step is required for the sEMG samples to be arranged before being supplied as input to the classifier. The enriched data from the previous phase is recorded for each designated muscle in a CSV file. For each of the five muscles, there are five CSV files. The normal samples are in the RF, BF, VM, ST, and FX folders. The aberrant knee samples are

Data Augmentation Algorithm (DAA)

Input:

- For the five considered muscles, preprocessed sEMG signal.

Output:

- new labeled signal with the same frequency domain and time characteristics

Steps:

1: **For each preprocessed sEMG signal for the five considered muscles**

2: According to the specified size, Define a new window.

3: Within the moving window, Compute RMS.

4: Smooth signal of sEMG sequences.

5: Remove all border effects.

6: Arrange the window movement.

7: Optimize looping movement.

8: Smooth a time window between 50 and 100ms more or less.

9: **End**

	0	1	2	3	...	69370	69371	69372	69373
0	0.452986	0.452339	0.451537	0.452663	...	0.0	0.0	0.0	0.0
1	0.725015	0.724097	0.723404	0.724260	...	0.0	0.0	0.0	0.0
2	0.871674	0.870337	0.872931	0.872189	...	0.0	0.0	0.0	0.0
3	0.868717	0.869153	0.868203	0.868639	...	0.0	0.0	0.0	0.0
4	0.448847	0.447602	0.447991	0.447337	...	0.0	0.0	0.0	0.0
5	0.956830	0.957963	0.955083	0.957988	...	0.0	0.0	0.0	0.0
6	0.471319	0.470101	0.469267	0.471006	...	0.0	0.0	0.0	0.0
7	0.901833	0.903493	0.900709	0.902959	...	0.0	0.0	0.0	0.0
8	0.719101	0.720545	0.718676	0.720710	...	0.0	0.0	0.0	0.0
9	0.445890	0.444050	0.445035	0.444379	...	0.0	0.0	0.0	0.0
10	1.000000	1.000000	1.000000	1.000000	...	0.0	0.0	0.0	0.0
11	0.266706	0.267614	0.267139	0.268639	...	0.0	0.0	0.0	0.0
12	0.460674	0.460628	0.459811	0.461538	...	0.0	0.0	0.0	0.0
13	0.463040	0.460628	0.460993	0.460947	...	0.0	0.0	0.0	0.0
14	0.500887	0.499704	0.496454	0.494675	...	0.0	0.0	0.0	0.0
15	0.415730	0.414446	0.414303	0.415385	...	0.0	0.0	0.0	0.0
16	0.000000	0.000000	0.000000	0.000000	...	0.0	0.0	0.0	0.0
17	0.438794	0.439313	0.437943	0.439053	...	0.0	0.0	0.0	0.0
18	0.860438	0.859680	0.859338	0.858580	...	1.0	1.0	1.0	0.0
19	0.441159	0.438721	0.438534	0.438462	...	0.0	0.0	0.0	0.0
20	0.447664	0.445826	0.446217	0.446746	...	0.0	0.0	0.0	0.0
21	0.441159	0.441681	0.440898	0.440828	...	0.0	0.0	0.0	0.0
22	0.962153	0.963292	0.960993	0.962722	...	0.0	0.0	0.0	0.0
23	0.389119	0.388396	0.388298	0.388757	...	0.0	0.0	0.0	0.0

Fig. 6. Knee abnormal samples for FX muscle.

	RF	BF	VM	ST	FX
0	0.0030	0.0045	-0.0158	-0.0008	3.0
1	0.0037	0.0045	-0.0165	-0.0008	3.0
2	0.0000	0.0037	-0.0173	-0.0015	3.0
3	-0.0008	0.0007	-0.0120	-0.0015	3.0
4	0.0000	0.0007	-0.0068	0.0007	3.0
...	...	...	...	...	...
14515	-0.0060	-0.0046	-0.0015	-0.0030	10.1
14516	-0.0060	-0.0046	-0.0015	-0.0030	10.1
14517	-0.0060	-0.0046	-0.0015	-0.0030	10.1
14518	-0.0060	-0.0046	-0.0015	-0.0030	10.1

Fig. 7. Raw Data Signal.

	RF	BF	VM	ST	FX
0	0.379139	0.391759	0.311905	0.366636	0.000000
1	0.384934	0.391759	0.303571	0.366636	0.000000
2	0.354305	0.390598	0.294048	0.366210	0.000000
3	0.347682	0.386245	0.357143	0.366210	0.000000
4	0.354305	0.386245	0.419048	0.367549	0.000000
...	...	...	...	...	...
14515	0.304636	0.378555	0.482143	0.365297	0.066542
14516	0.304636	0.378555	0.482143	0.365297	0.066542
14517	0.304636	0.378555	0.482143	0.365297	0.066542
14518	0.304636	0.378555	0.482143	0.365297	0.066542

Fig. 8. The Output Rectified Signal.

contained in five additional files for each of the five muscles. In developing the training model, a labelling field is added, with 1 for knee patients and 0 for normal instances. The normal and pathological knee samples are concatenated for each of the five muscles, as shown in Fig. 10.

The data were shuffled using randomly permuted indices to create a balanced training model. The resulting training cases are fed into real-time classification algorithms as input data. A five-dimensional matrix representing the full dataset for real-time classifiers is created by concatenating normal and pathological knee sample data for each of the five muscles, with each row representing a single input, as shown in Fig. 11. The algorithm for concatenating data and shuffling steps is given by Algorithm 3.

Data Concatenation and Shuffling Algorithm (DCSA)

Input:

- *For the five muscles, input both normal and abnormal sEMG samples.*

Output:

- *sEMG training model for the five muscles.*

Steps:

1: **For the five muscles, input both normal and abnormal sEMG samples**
2: Label the normal samples as 0, and knee patients as {1} and
3: For each of the data samples
4: Concatenate data samples.
5: Obtain permuted indices randomly.
6: Using the randomly permuted indices, shuffle the data.
7: Next
8: For each muscle, copy the data to a CSV file.
9: **End**

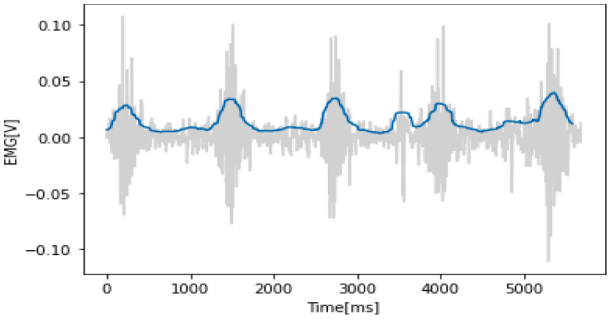


Fig. 9. The Output Rectified Signal.

3.4. Lower limb classification (LLC)

The lower limb classification phase was performed using three deep feedforward networks; RNN, LSTM, and GRU. The three deep network models are suitable for processing the sequence data in terms of the sEMG signal for the knee. Inherent ability to learn sequentially for the three comparative time series deep networks; timestep was measured. The three comparative deep networks; have the inherent ability to learn sequentially from time series data.

0	0.661905	0.608355	0.566474	0.569191	0.558824	0.665888	0.740741	0.797531	0.848485	0.75	0.770992	0.497436	0.306358	0.577465	0.40107	0.181598	0.266667	0.690083	0.799216
0	0.733333	0.665796	0.523121	0.509138	0.542986	0.595794	0.740741	0.683951	0.666667	0.583333	0.568702	0.266667	0.086705	0.739437	0.561497	0.181598	0.361702	0.759298	0.887843
0	0.75	0.686684	0.566474	0.569191	0.61086	0.735981	0.851852	0.925926	0.939394	0.875	0.942748	0.882051	0.780347	0.894366	0.518717	0.164649	0.287943	0.721074	0.82902
0	0.642857	0.530026	0.479769	0.509138	0.524887	0.649533	0.740741	0.777778	0.818182	0.666667	0.687023	0.651282	0.566474	0.633803	0.320856	0.09201	0.24539	0.71281	0.823529
0	0	0	0.066474	0.255875	0.38914	0.579439	0.740741	0.814815	0.848485	0.75	0.629771	0.266667	0.086705	0.15493	0	0.01937	0	0	0
0	0.642857	0.548303	0.5	0.548303	0.61086	0.649533	0.760494	0.834568	0.832323	0.791667	0.744275	0.574359	0.520231	0.633803	0.641711	0.237288	0.351773	0.752066	0.840784
0	0.62619	0.608355	0.543353	0.569191	0.576923	0.579439	0.68642	0.760494	0.694949	0.647222	0.568702	0.302564	0	0	0.037433	0	0.222695	0.673554	0.80549
0	1	1	1	1	1	0.824766	0.649383	0.407407	0.242424	0	0	0	0.346821	0.84507	0.962567	0.418886	0.351773	0.728306	0.793725
0	0.411905	0.313316	0.306358	0.469974	0.558824	0.700935	0.851852	0.945679	0.953535	0.916667	0.942748	0.805128	0.566474	0.84507	0.759358	0.309927	0.382979	0.783058	0.882353
0	0.62619	0.490862	0.479769	0.409922	0.576923	0.665888	0.797531	0.908642	0.80202	0.855556	0.71374	0.497436	0.479769	0.528169	0.481283	0.145278	0.30922	0.728306	0.82902
0	0.642857	0.548303	0.566474	0.569191	0.644796	0.771028	0.851852	0.908642	0.80202	0.75	0.744275	0.615385	0.479769	0.577465	0.481283	0.053269	0.234043	0.705579	0.823529
0	0.5	0.430809	0.413295	0.430809	0.457014	0.544393	0.68642	0.760494	0.771717	0.855556	0.973282	0.692308	0.566474	1	0.882353	0.382567	0.394326	0.774793	0.870588
0	0.642857	0.530026	0.5	0.490862	0.524887	0.579439	0.68642	0.703704	0.680808	0.602778	0.599237	0.497436	0.393064	0.471831	0.40107	0.181598	0.340426	0.728306	0.840784
0	0.733333	0.665796	0.523121	0.509138	0.542986	0.595794	0.740741	0.683951	0.666667	0.583333	0.568702	0.266667	0.086705	0.739437	0.561497	0.181598	0.361702	0.759298	0.887843
1	0.20404	0.256757	0.175992	0.129293	0.11847	0.09585	0.053333	0.143333	0.144578	0.230824	0.151163	0.203313	0.173356	0.285585	0.377208	0.379048	0.349206	0.291824	0.327037
1	0.082828	0.094595	0	0	0	0	0.015385	0.175	0.204819	0.301075	0.255814	0.316265	0.321093	0.448776	0.569788	0.628571	0.643386	0.574843	0.548996
1	0.09798	0.175676	0.114954	0.083838	0.083022	0.066206	0.046154	0.175	0.210442	0.327599	0.267442	0.310994	0.282664	0.347235	0.443463	0.442857	0.412698	0.281761	0.20307
1	0.166667	0.22973	0.175992	0.136364	0.132463	0.118577	0.092308	0.218333	0.240964	0.316846	0.250388	0.316265	0.314261	0.407978	0.496466	0.5	0.492063	0.415094	0.371901
1	0.242424	0.283784	0.229908	0.19697	0.195896	0.184783	0.184615	0.275	0.289157	0.359857	0.302326	0.356175	0.333903	0.42883	0.496466	0.5	0.484656	0.386164	0.336482
1	0.234343	0.256757	0.191251	0.159596	0.146455	0.148221	0.153846	0.3	0.301205	0.365591	0.296899	0.356175	0.314261	0.394379	0.477032	0.479048	0.45291	0.367296	0.371901
1	0.242424	0.276577	0.19939	0.136364	0.11194	0.103755	0.085128	0.205833	0.240964	0.327599	0.262016	0.322289	0.295474	0.38078	0.483216	0.485714	0.435979	0.32956	0.299882
1	0.181818	0.236036	0.160732	0.136364	0.132463	0.118577	0.107692	0.225	0.26506	0.316846	0.250388	0.322289	0.327071	0.435177	0.490283	0.514286	0.484656	0.386164	0.354191
1	0.181818	0.195495	0.114954	0.114141	0.125933	0.118577	0.085128	0.2125	0.283534	0.333333	0.226357	0.287651	0.282664	0.388033	0.449647	0.493333	0.47619	0.405031	0.397875

Mus1_RF

Fig. 10. Concatenated and labeled data for each of the five muscles.

1	0.611402	0.596732	0.584422	0.531715	0.515152	0.471111	0.402062	0.394376	0.452411	0.470972	0.350474	0.33193	0.310345	0.275	0.309216	0.296703	0.211836	0.16984	0.236842
1	0.058824	0.03827	0.235382	0.40619	0.476834	0.325843	0.274444	0.211538	0.211864	0.269231	0.5666	0.700499	0.784648	0.79476	0.621138	0.761111	0.899058	0.5625	0.19877
0	0.107086	0.10699	0.10724	0.105102	0.105864	0.108369	0.105	0.106232	0.10724	0.103066	0.109119	0.104063	0.10408	0.105331	0.103963	0.103845	0.105323	0.102941	0.107103
1	0.216667	0.681609	0.91258	0.944167	0.965494	0.921817	0.787037	0.704531	0.483871	0.153457	0.042063	0.039238	0.112676	0.530026	0.671533	0.90303	0.958988	0.969008	0.517964
1	0.498379	0.528933	0.636458	0.651174	0.661075	0.696667	0.718738	0.734206	0.781439	0.824294	0.856723	0.871124	0.879985	0.882329	0.894444	0.87807	0.716673	0.743823	0.811993
1	0.059196	0.058189	0.058871	0.059089	0.0583	0.060497	0.05669	0.059639	0.058276	0.060808	0.060157	0.059099	0.059117	0.05904	0.058969	0.059903	0.05904	0.060878	0.06022
0	0.059034	0.061826	0.058861	0.062742	0.060597	0.06069	0.061507	0.060759	0.060309	0.061608	0.059453	0.058689	0.062359	0.057445	0.062133	0.059221	0.060203	0.060271	0.06015
1	0.776471	0.758715	0.757576	0.730268	0.720041	0.784156	0.818599	0.724138	0.714446	0.543961	0.523933	0.59276	0.702782	0.608769	0.636804	0.525926	0.272727	0.140741	0.088785
1	0.556735	0.74894	0.63973	0.700807	0.808944	0.784462	0.755878	0.57235	0.50546	0.509494	0.526923	0.542221	0.508333	0.515334	0.49666	0.422238	0.22452	0.06383	0.077451
1	0	0.063228	0.212894	0.288201	0.247104	0	0	0.009615	0.021469	0	0	0.362729	0.624733	0.712882	0.572358	0.770833	0.919246	0.658333	0.368852
0	0.5	0.490862	0.524887	0.579439	0.68642	0.703704	0.680808	0.602778	0.599237	0.497436	0.393064	0.471831	0.40107	0.181598	0.340426	0.728306	0.840784	0.829545	0.797733
1	0.265306	0.099915	0.105928	0.076235	0.213994	0.325255	0.36	0.577656	0.721154	0.803475	0.898321	0.839975	0.870905	0.852424	0.834427	0.852896	0.945146	1	1
1	0.406933	0.362319	0.370968	0.434314	0.532101	0.601805	0.404006	0.279912	0.588235	0.838944	0.84313	0.741379	0.807352	0.771773	0.766772	0.457544	0.111271	0.063559	0.11131
0	0.605263	0.564558	0.385682	0.406548	0.400172	0.458435	0.573171	0.742786	0.754395	0.70929	0.476208	0.523874	0.846939	0.839939	0.883439	0.954163	0.978922	1	0.97554
1	0.725473	0.719171	0.658113	0.775367	0.892157	0.912688	0.928079	0.976242	1	0.974271	0.939547	0.93975	0.938889	0.882365	0.820426	0.721239	0.5375	0.105724	0.376884
1	0.114954	0.083838	0.083022	0.066206	0.046154	0.175	0.210442	0.327599	0.267442	0.310994	0.282664	0.347235	0.443463	0.442857	0.412698	0.281761	0.20307	0.241249	0.282353
1	1	1	1	1	0.882353	0.76574	0.516854	0.40095	0.546502	0.896257	1	0.994475	0.96431	0.924171	0.751464	0.910383	0.79173	0.626335	0.689462
0	0.45348	0.605962	0.674653	0.776928	0.772998	0.532619	0.200321	0.334112	0.34375	0.452436	0.479817	0.298851	0.335035	0.355081	0.333333	0.346173	0.380597	0.413828	0.442517
1	1	1	0.950331	0.497766	0.078431	0	0	0	0	0	0	0	0	0	0	0	0.090417	0.413468	1
1	0.846715	0.747748	0.676504	0.548521	0.484076	0.50638	0.506849	0.504374	0.464589	0.402206	0.578859	0.875737	0.90548	0.924498	0.860144	0.744774	0.5894	0.575654	0.694802
0	0.059239	0.058387	0.059189	0.059245	0.058173	0.060277	0.05643	0.059364	0.058325	0.061039	0.060199	0.059169	0.059336	0.059273	0.059286	0.06022	0.059223	0.061005	0.060242
1	0.570825	0.591919	0.470395	0.454264	0.50536	0.340587	0.226453	0.315878	0.36105	0.565276	0.707804	0.822303	0.737705	0.532872	0.301508	0	0.152284	0.368737	0.227656
0	0.607547	0.636364	0.578261	0.58356	0.433047	0.50892	0.590116	0.632959	0.818744	0.789002	0.92725	0.898734	0.712092	0.442755	0.195402	0.108508	0.184167	0.476744	0.459653

M1_RF - Copy

Fig. 11. Balanced training data.

3.4.1. Recurrent neural networks (RNNs)

The RNN input sequence is represented in vectors. The vectors' size is determined by the timestep and input features. It iteratively cycles through timesteps, combining its current state and the size input features to produce the output at each timestep. The state of the RNN is reset between the processing of two independent sequences, so one sequence is still considered a single input to the network. This data point is handled in a single step instead of looping the network over the sequence elements, as shown in Fig. 12 [32].

In our case, the final result is a 2D vector of timesteps and output features, with each timestep representing the loop's current output. The following overview of the whole state sequence is obtained by training a basic recurrent network with an embedding layer and standard RNN layer, as shown in Algorithm 4.

Tuning Parameters Algorithm (TPA)

Input:

- A three-dimensional tensor with the shape [batch, timesteps, feature].

Output:

- Tuned Parameters

Steps:

1: Initial state: The starting state of the tensors to be supplied to the cell's first call.

2: Units: The output space's dimensionality.

3: Activation: To employ the activation function. Hyperbolic tangent is the default (tanh).

4: Use bias: A Boolean variable (default True)

3.4.2. Long Short-Term memory (LSTM) (Redundant)

LSTM is characterized by additional data flows of information across timesteps for representing the difficult case that occurred via a carry track C_t denotes carry, and the values at different timesteps are called C_t [33]. This data is integrated with the input and recurrent connections via a dot product with a weight matrix, bias, and the use of an activation function. With the help of a multiplication operation and an activation function, this will affect the state that is passed on to the next timestep. The carry dataflow is a mechanism to modify the following output and state from a conceptual standpoint, as seen next. Fig. 13 depicts the anatomy of an LSTM [11].

This is what the interpretation of these procedures is supposed to accomplish. Multiplying c_t and f_t, for example, is a method of erasing irrelevant data from the carried data flow. In the meantime, i_t, and o_t, give current information, updating the carry track with new data [14].

3.4.3. Gated Recurrent Unit (GRU) redundant

Using SGD and early stopping-based classification loss, the networks are trained. Early stopping during data training reduces the models overfitting, although hyper-parameter adjustment is also necessary to get a suitable local solution [13]. The network architecture's size,

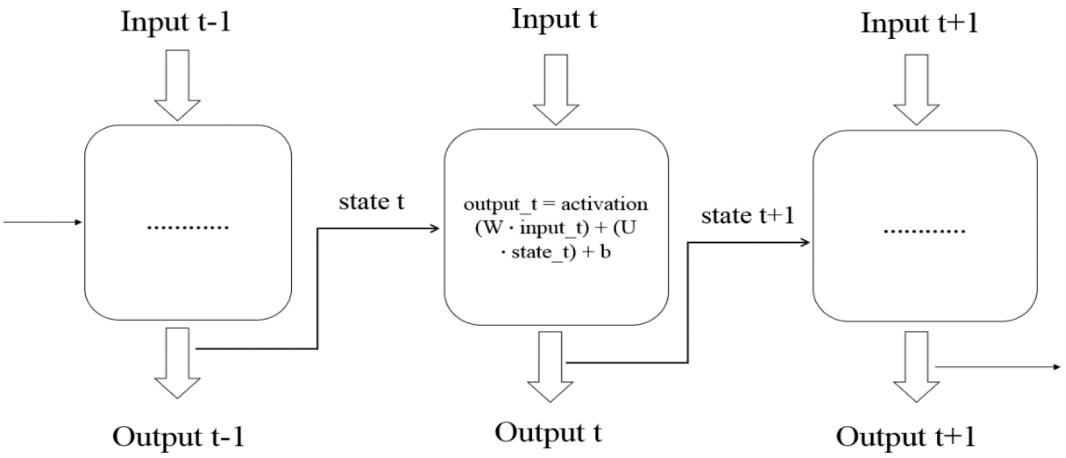


Fig. 12. The entire sequence of basic RNN.

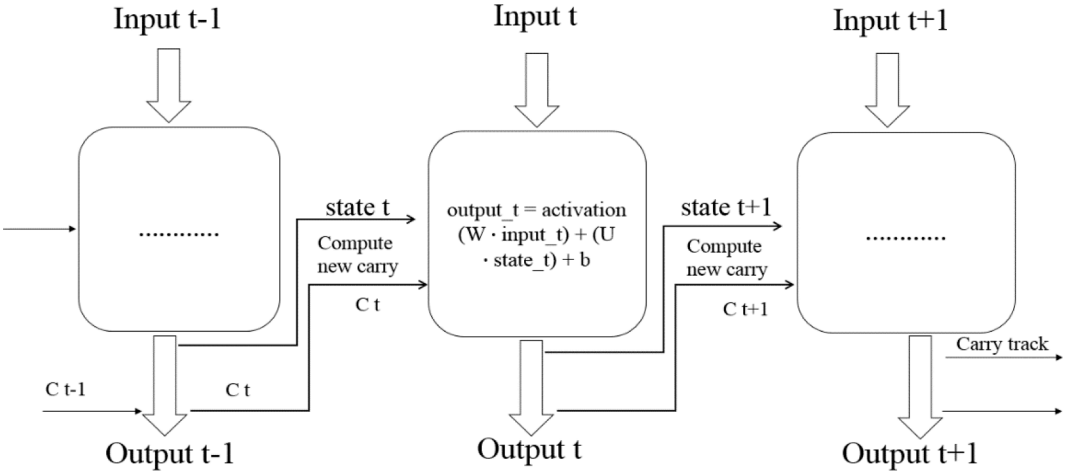


Fig. 13. The anatomy of LSTM.

number of layers, learning rate, batch size, momentum, and sequence length are the hyper-parameters of importance. Trials or stochastic searches are typically used to set tuning hyper-parameters. It is recommended over manual search since LSTM networks are resource-intensive. While other parameters are set to standard settings, the architecture and learning parameters need user adjustment.

4. Results and discussion

This section introduces the dataset that was utilised, the performance indicators, and the algorithm's outcomes.

4.1. Dataset description

This database contains samples from 11 healthy people and 11 who had a knee abnormality diagnosed by a specialist [34]. Electromyography and goniometry equipment, as well as MWX8 Datalog Biometrics, were used to get the data. Eight digital and four analog channels were employed by the MWX8 data log, with four for sampling and one for goniometry. Using a microSD card, this data was directly stored on the computer's internal storage and then sent through Bluetooth to software with real-time datalogging with a 14-bit resolution and a 1000 Hz sample rate.

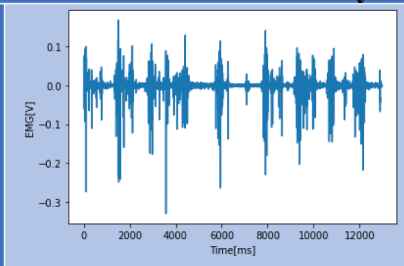
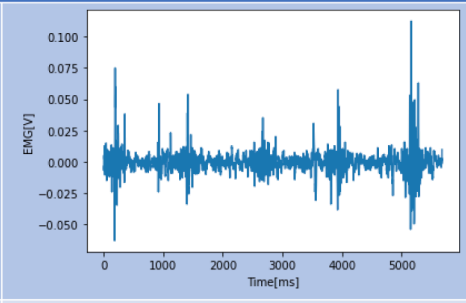
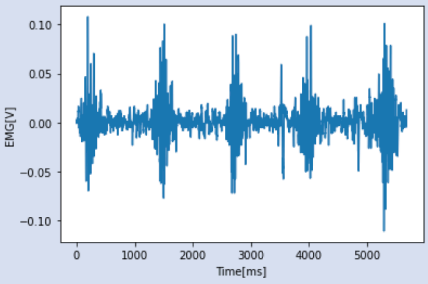
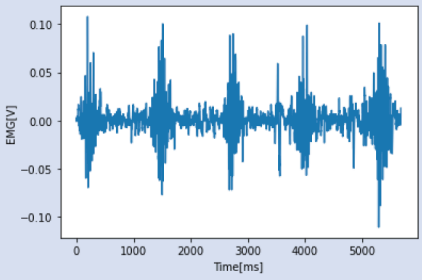
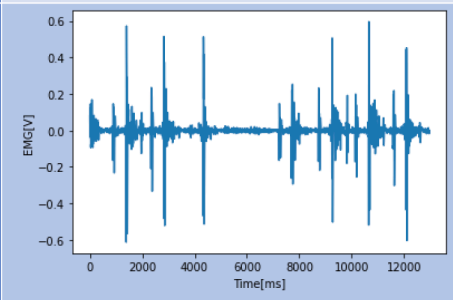
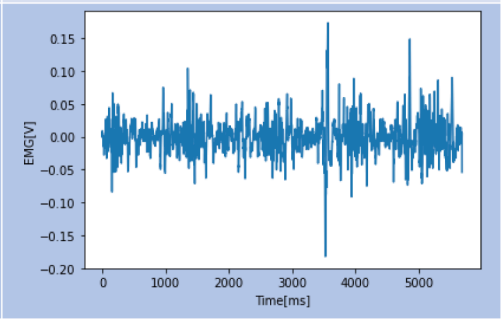
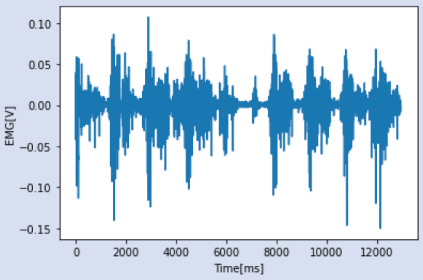
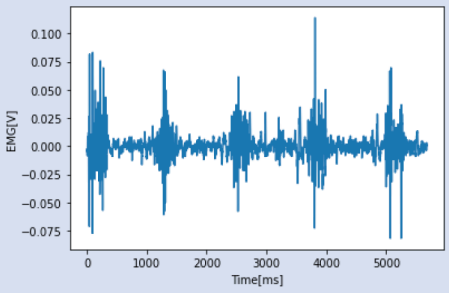
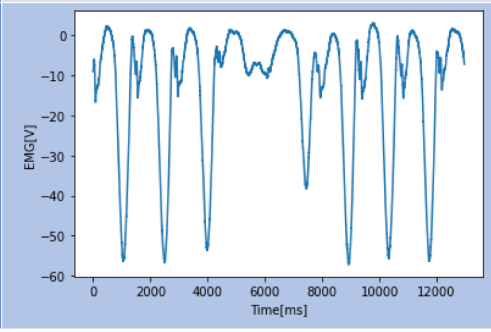
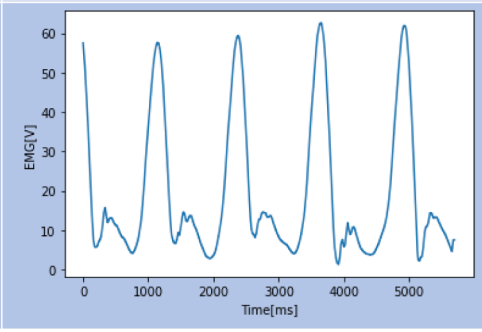
The 22 participants in our study underwent three actions to examine behaviour related to the knee muscle: leg flexion up, leg extension from a sitting posture, and walking. The data was collected using four electrodes (Vastus Medialis, Biceps Femoris, Rectus Femoris, and Semitendinosus) and a knee goniometer. The total number of electrodes is four, one for each channel in the time series (1–4). Each subject has five shares or motion repeats in each series. Each data file has five columns, organized as shown in Table 1.

Table 1
An overview of the Lower Limb dataset.

Segment	Lower Limb				
Channel	Ch1	Ch2	Ch3	Ch4	Ch5
Muscle	RF	BF	VM	ST	FX
Column	0	1	2	3	4

The segment denotes the location on the body where the data is collected, and the channel denotes the electrode linked to a muscle. Recto Femoral (RF), Semitendinosus (ST), Vastus Medialis (VM), Biceps Femoral (BF), and Flexion (FX) of the Knee are all abbreviated as RF, BF, VM, ST, and FX, as shown in Table 2. Each subject is photographed in three ways: sitting, standing, and walking. The training database has 66 records for knee abnormalities along with normal ones, with five attributes describing the corresponding muscles measured. The dataset was augmented using a time-series generator that yielded samples and their targets. The augmentation step resulted in 1056 records of EMG sequence signals for the five corresponding muscles.

Table 2
Normal and Knee Abnormality samples for five core muscles.

Muscles	Knee Abnormality	Normal
BF		
RF		
VM		
ST		
FX		

4.2. Performance metrics

The model's performance is evaluated via four different performance indicators using a 10-fold cross-validation technique. The performance indicators are ACC, Loss, Validation Acc, and Validation Loss.

- ACC is calculated as the percentage of accurate predictions for all samples by Equation (4) [35]. The ACC is determined by the following equation in terms of these arguments, true positive (TP), false positive (FP), true negative (TN), and false negative (FN) [36]. TP shows that the predicted cases and their actual output are true,

while TN demonstrates the falsity of the expected cases and their real output. The predicted cases are confirmed by FP, but the actual output is demonstrated to be erroneous. Finally, FN demonstrates that while the actual output is correct, the predicted cases are false. Arguments are calculated for knee abnormalities against normal ones [37].

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

- The loss function is a measure of model prediction in terms of predicting the expected outcomes. It maps the values of one or more variables onto a real case associated with the study. It includes various terms around the difference between determined and real-world values for data samples. The objective is either to minimize a loss function or its opposite, which is to be maximized [38]. The loss function can be calculated based on a quadratic function, commonly called the Mean Squared Error (MSE) technique. The quadratic function is particularly good because of its variance properties and because it is symmetric—the error above the target results in a loss value equivalent to the error below the target. If the target is t, then a quadratic loss function f(x) is given by Equation (5) [36].

$$f(x) = c\,(t - x)^2 \tag{5}$$

where C is a constant, that makes no difference and can be ignored by setting its value to 1.

The loss function can also be determined based on the cross-entropy. They both compute the difference between the probability that occurs and the probability that is predicted. Cross-entropy loss employs negative log probability for the outputs, whereas cross-entropy still uses log probability, as given by Equation (6) [39].

$$f(x) = -\log(p(x)) \tag{6}$$

A softmax function can be used to easily convert log probabilities into normal numbers for computation. When using cross-entropy loss instead of mean squared error loss, models with sigmoid and softmax outputs perform better than those that suffer from saturation and poor learning. The forecast of a short-term vs. long-term prediction model is contrasted in time series classification problems using cross-entropy loss. When one class has a significantly higher probability, cross-entropy loss results in a significant penalty [40].

- The F1 score is a metric used to evaluate the accuracy of a test. It takes into account both precision and recall values obtained from the test. Precision measures the proportion of true positive results among all positive predictions, while recall measures the proportion of true positive results among all actual positive samples. The F1 score is calculated as given by Equation (7) [16]:

$$F1 = \frac{2\,x\,TP}{(2\,x\,TP + FP + FN)} \tag{7}$$

- Precision, which is also known as positive predictive value, is a metric used to assess the accuracy of a test. It is calculated by dividing the number of true positive results by the total number of positive predictions, including false positives. The formula for precision is given by Equation (8) [1]:

$$Precision = \frac{TP}{(TP + FP)} \tag{8}$$

- AUC stands for "Area Under the ROC Curve". It is a metric that quantifies the overall performance of a model by measuring the two-dimensional area beneath the entire ROC curve, from (0,0) to (1,1). AUC provides a comprehensive evaluation of the model's performance across all classification thresholds. It can be interpreted as the probability that the model will rank a random positive case higher than a random negative case [16].

4.3. Experimental results

The proposed framework is implemented using MATLAB 2017 on an HP Envy laptop with an AMD FX CPU for concatenation and shuffling. Cloud computing (Google Colab) is used for augmentation and classification steps, and Colab is a GPU-centric application for accelerating deep learning. The runtime hardware configuration to execute the model was 12 GB of RAM, the Nvidia K80 GPU, and 2496 CUDA cores.

Four performance measures, accuracy, loss, validation accuracy, validation loss, recall, precision, and F1-score are used to evaluate the model's performance via 10-fold cross-validation. A comparative experiment is performed for the three deep learning networks for time series and sequence data. The commonly used deep learning techniques for time series and sequence data are simple RNN, LSTM, and GRU. The deep networks are suitable for input data sequences regarding sEMG sequence signals for the corresponding five muscles. For continuous estimation of the knee signal, the utilized deep techniques process the sEMG input sequences by iterating through the sequence elements and maintaining a state that contains information about what they have seen. The summary of the results in terms of ACC and loss is given in Table 3 and Table 4 for each of the five muscles, RF, BF, VM, ST, and FX,

Table 3
The performance evaluation for accuracy for the three comparative deep networks.

Recurrent Net	Accuracy				
	RF	BF	VM	ST	FX
Simple RNN	0.961	0.951	0.915	0.938	0.834
LSTM	0.967	0.957	0.926	0.941	0.847
GRU	0.976	0.963	0.945	0.945	0.857

Table 4
The performance evaluation for loss for the three comparative deep networks.

Recurrent Net	Loss				
	RF	BF	VM	ST	FX
Simple RNN	0.154	0.192	0.212	0.176	0.156
LSTM	0.147	0.176	0.264	0.157	0.508
GRU	0.111	0.157	0.218	0.288	0.461

Table 5
The performance evaluation for validation accuracy for the three comparative deep networks.

Recurrent Net	Validation Accuracy				
	RF	BF	VM	ST	FX
Simple RNN	0.943	0.934	0.897	0.934	0.687
LSTM	0.945	0.941	0.903	0.931	0.775
GRU	0.949	0.944	0.939	0.936	0.787

Table 6
The performance evaluation for validation loss for the three comparative deep networks.

Recurrent Net	Validation Loss				
	RF	BF	VM	ST	FX
Simple RNN	0.244	0.223	0.171	0.163	0.213
LSTM	0.217	0.220	0.175	0.170	0.631
GRU	0.212	0.216	0.179	0.135	0.687

respectively.

The summary of the results in terms of validation accuracy and validation loss is given in Tables 5 and 6 for each of the five muscles, RF, BF, VM, ST, and FX, respectively.

Table 7, 8, and 9 display the percentage of achievement for each of the five muscles (RF, BF, VM, ST, and FX) based on metric standards such as Recall, Precision, and F1-score, providing insight into the efficiency of the proposed system.

To showcase deep networks' sequential learning ability, the three comparative time series were compared using time steps. Time steps refer to the number of input variables (X) utilized to predict the next time step (Y), thus requiring the removal of several initial dataset rows for each time step in the representation. Table 10 provides a summary of the results for each of the five muscles (RF, BF, VM, ST, and FX) in terms of time steps as a measure of sequence length.

Table 11 takes into account the training time when assessing the optimal time series for deep networks.

Fine-tuning hyperparameters for each three-time series deep network model on the sEMG knee dataset during training experiments and on the validation datasets whenever necessary. The following table lists all hyperparameters for RNN, LSTM, and GRU architectures in terms of the first hidden layer of a network (the embedding layer) and the number of hidden layers in the LSTM layer for learning long-term dependencies between sequence data and time steps in time series. The weighted sum of the input is processed into an output according to the activation function. The optimizer that is used in optimizing and minimizing the loss function (loss function calculation, number of epochs, and number of training instances) is used in a single iteration (batch size) and validation split operator. Table 12 shows the hyperparameters used in training experiments with the three-time series deep network model on the sEMG knee dataset.

The predictive model's effective performance was demonstrated by comparing it with other state-of-the-art models that used the same datasets. A comparison of the used deep learning models vs. previous algorithms are given in Tables 13 and in Fig. 14 for each of the five

Table 7
The performance evaluation for Recall for the three comparative deep networks.

Recurrent Net	Recall				
	RF	BF	VM	ST	FX
Simple RNN	1.0	0.991	0.995	1.0	0.9571
LSTM	0.997	1.0	0.926	0.991	0.974
GRU	0.996	0.963	1.0	0.995	0.897

Table 8
The performance evaluation for Precision for the three comparative deep networks.

Recurrent Net	Precision				
	RF	BF	VM	ST	FX
Simple RNN	1.0	0.981	0.996	1.0	0.967
LSTM	1.0	0.995	0.962	0.981	0.974
GRU	0.998	0.953	0.997	0.985	0.899

Table 9
The performance evaluation for F1-score for the three comparative deep networks.

Recurrent Net	F1-score				
	RF	BF	VM	ST	FX
Simple RNN	0.899	0.991	0.996	0.992	0.927
LSTM	0.927	0.979	0.962	0.971	0.934
GRU	0.938	0.958	0.989	0.964	0.896

Table 10
The performance evaluation of timestep/sequence lengths for the three comparative deep networks.

Recurrent Net	time-steps/sequence lengths				
	RF	BF	VM	ST	FX
Simple RNN	426 ms/ step	428 ms/ step	428 ms/ step	325 ms/ step	322 ms/ step
LSTM	430 ms/ step	438 ms/ step	424 ms/ step	505 ms/ step	530 ms/ step
GRU	443 ms/ step	428 ms/ step	427 ms/ step	504 ms/ step	329 ms/ step

Table 11
The performance evaluation of train running time for the three comparative deep networks.

Recurrent Net	Training time				
	RF	BF	VM	ST	FX
Simple RNN	68 s	74 s	81 s	70 s	60 s
LSTM	74 s	76 s	84 s	73 s	66 s
GRU	77 s	78 s	87 s	75 s	62 s

Table 13
Used DL models vs. previous algorithms.

Recurrent Net	Validation Accuracy				
	RF	BF	VM	ST	FX
Simple RNN	0.943	0.934	0.897	0.934	0.687
LSTM	0.945	0.941	0.903	0.931	0.775
GRU	0.949	0.944	0.939	0.936	0.787
Ibraheem et al. [15]	0.841	0.840	0.838	0.837	0.821
Bansal et al. [16]	0.857	0.858	0.850	0.849	0.841

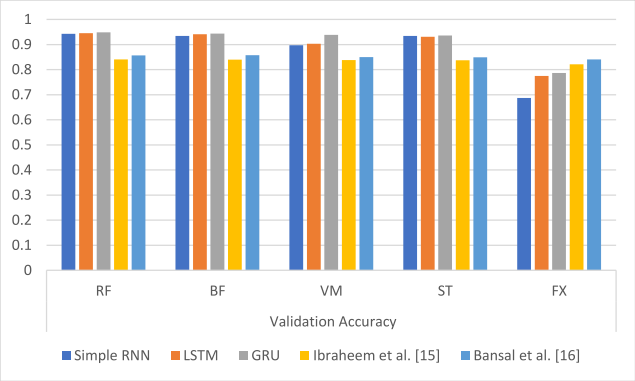


Fig. 14. Used DL models vs. previous algorithms.

Table 12
Fine-tuning the hyperparameters in training the three comparative deep networks.

Recurrent Net	Hyperparameters							
	Embedding	LSTM	Activation	Optimizer	Loss	Epochs	Batch size	Validation split
Simple RNN	(1000, 32)	32	'sigmoid'	'rmsprop'	'binary_crossentropy'	30	128	0.2
LSTM	(1000, 128)	32	'sigmoid'	'adam'	'binary_crossentropy'	30	128	0.2
GRU	(1000, 128)	32	'sigmoid'	'rmsprop'	'mse'	30	128	0.2

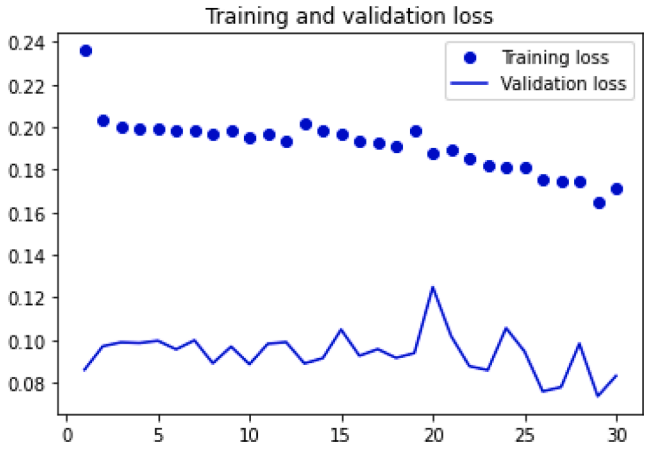


Fig. 15. The training and validation loss curves over epochs for RNN.

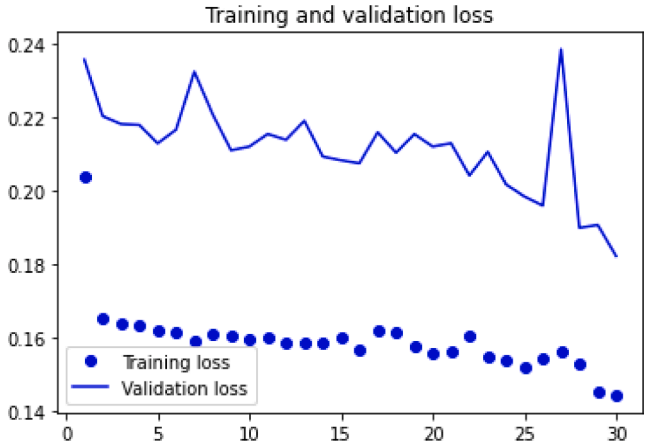


Fig. 16. The training and validation loss curves over epochs for LSTM.

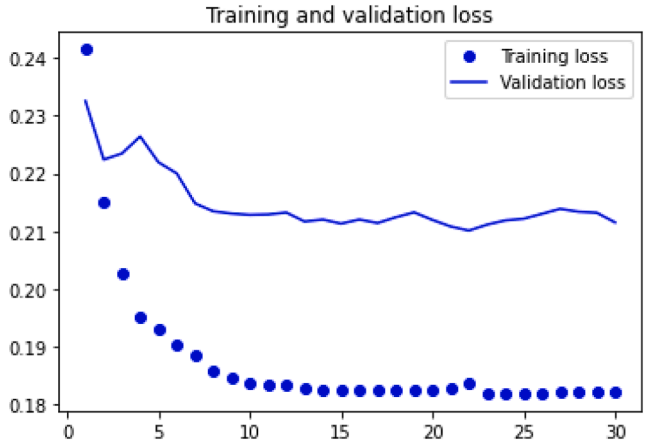


Fig. 17. The training and validation loss curves over GRU epochs.

muscles, RF, BF, VM, ST, and FX, respectively.

As illustrated in Table 13, it compares the performance results of various recurrent neural network (RNN) models for a given task, as measured by validation accuracy. The models include Simple RNN, LSTM, and GRU, as well as two previously published models by Ibraheem et al. and Bansal et al. The results indicate that the GRU model performs the best overall, with a validation accuracy of 0.949. The LSTM model also performs well, with a validation accuracy of 0.945. The Simple RNN model performs moderately well, with a validation accuracy ranging from 0.687 to 0.943 depending on the variant. The models by Ibraheem et al. and Bansal et al. perform less well, with validation accuracies ranging from 0.837 to 0.858. These results suggest that the use of RNN models, particularly GRU and LSTM, can be effective for this task, and that careful selection of the specific RNN architecture can have a significant impact on performance.

4.4. Discussion

The three deep learning networks we used in this paper look like feedforward neural networks, with one difference: they have connections pointing backward. The recurrent layer gets the input x(t) and the output from the previous time step at each time step t. Each recurrent neuron in the network has two weight sets, one for the input at the present time step (wx) and the other for the previous time's outputs. This time series technique somehow replaced the need for a feature extraction step, saving time and effort. As mentioned previously, the proposed technique's primary goal is to diagnose knee abnormalities based on analyzing the sEMG signals. The system is based on the idea that deep learning techniques can automatically extract features from the given data. We used two deep learning techniques: RNN, LSTM, and GRU, to compare the results and use the one with the best results. We tuned the parameter settings to achieve the best accuracy. The parameters we used are listed in Table 8.

The experimental results show that GRU gives us the best results. On the other hand, we applied an electrode to five different muscles in the knee, and the extracted signals were detected to be used as input to the deep learning network. The tables above show the lower limb deep model classification accuracy and loss using different time series and deep sequence networks for sEMG for each of the five knee muscles using 10-fold cross-validation. Tables 3, 4, and 5 show that the best results are obtained using the GRU network and show higher results with the signals from RF, RF, BF, VM, ST and FX, respectively. Table 6 shows that the best results are obtained using the GRU network and show higher results with the signals from FX, ST, BF, VM, RF and BF, respectively.

The results of the comparative study show that the GRU sequence deep network model outperforms other time series deep network techniques. RNN is commonly used due to its simplicity in real-world applications. But, RNN needs to remember information about inputs viewed many timesteps before; at time t, it is difficult to learn such long-term dependencies in practise. Thus, due to the vanishing gradient problem with many-layer non-recurrent networks (feedforward networks), the network becomes untrainable as the number of layers in the network grows.

LSTM and GRU layers conquer this issue by stacking the data from the grouping onto the transport line whenever it is conveyed to a later timestep and, afterward, can be recovered depending on the situation. The LSTM accomplishes this by preserving information for future use and preventing the loss of older signals. GRU layer works on the same premise as LSTM but is more streamlined and thus less expensive to run. Table 7 confirms the ability of GRU to learn sequentially using min time steps more than that for the three comparative time series deep networks; each timestep appears to have min values for the signals from VM, BF, FX, and RF to ST, respectively.

For the calculation time comparison, Table 8 shows that RNN achieves training in less time than LSTM and GRU as the number of layers in a network grows. LSTM and GRU layers in a neural network can store information from a sequence and pass it along to future time steps, allowing for long-term dependencies to be captured, albeit at the cost of increased computational complexity. The training and validation loss curves are constructed to visualize the performance of the three comparative time-series deep network models. Fig. 15 displays the loss curves for training and validation over epochs for an RNN. Fig. 16 displays the loss curves for training and validation over epochs for LSTM.

Fig. 17 displays the training and validation loss curves over epochs for GRU.

The comparison of the lower limb GRU sequence deep network model with other sequence deep network models (RNN and LSTM) demonstrated its promising results. Fig. 17 illustrates that the evaluation score consistently outperforms the scores obtained with RNN and LSTM in Figs. 16 and 15, respectively.

5. Conclusion

A non-invasive time-sequence model has been proposed to accurately characterize muscle activity based on sEMG signals. The proposed deep learning model is designed to handle time-sequential data by controlling the input flow and avoiding the vanishing gradient problem. Three feedforward deep learning networks - RNN, LSTM, and GRU - have been used to represent and classify time series and sequence data. While RNN struggles with learning long-term dependencies, LSTM and GRU layers can retain information from the sequence at any time and retrieve it later. GRU layers have fewer parameters, superior performance, and consume less memory than LSTM layers, making them faster and more efficient in practice. For larger datasets and lengthy sequences, LSTM is preferred due to its higher accuracy. However, GRU is better in terms of memory efficiency and quicker results. The proposed deep network model using the GRU sequence for the five knee muscles shows promising results compared to RNN and LSTM models. The evaluation scores are consistently higher and more stable, indicating the model's effectiveness in accurately and timely interpreting PF OA. It can be a valuable tool for physicians.

Funding Statement

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

CRediT authorship contribution statement

Mai Ramadan Ibraheem: Conceptualization, Methodology, Supervision, Data curation, Software, Writing – original draft. **Saleh Naif Almuayqil:** Methodology, Writing – review & editing. **A. A. Abd El-Aziz:** Conceptualization, Supervision, Methodology, Writing – review & editing. **Medhat A. Tawfeek:** Conceptualization, Supervision, Methodology, Writing – review & editing. **Fatma M. Talaat:** Methodology, Conceptualization, Data curation, Resources, Supervision, Writing – original draft.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Ramadan Ibraheem M, adel J, Eldin Balbaa A, El-Sappagh S, Abuhmed T, Elmogy M. Timing and classification of patellofemoral osteoarthritis patients using fast large margin classifier. Comput Mater Continua 2021;67(1):393–409.
[2] Adel J, Balbaa AE, R. ElHabashy H, Ibraheem MR, Fayaz N, Abbass S, et al. Temporal activation of core muscles and vasti in isolated patellofemoral osteoarthritis during stair-stepping: a case-control study. Ann Appl Sport Sci 2022; 10(1).
[3] Ma Y, Donati E, Chen B, Ren P, Zheng N, Indiveri G, "Neuromorphic Implementation of a Recurrent Neural Network for EMG Classification", Aug. 2020.
[4] Azhiri RB, Esmaeili M, Nourani M, "Real-Time EMG Signal Classification via Recurrent Neural Networks", IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Houston, TX, USA, pp. 2628-2635, 2021.
[5] Simão M, Neto P, Gibaru O. EMG-based online classification of gestures with recurrent neural networks. Pattern Recogn Lett Dec. 2019;128:45–51.
[6] DiPietro R, Hager GD. Deep learning: RNNs and LSTM. Academic Press: Handbook of medical image computing and computer assisted intervention; 2020. p. 503–19.
[7] Wu X, Wang H-Y, Shi P, Sun R, Wang X, Luo Z, et al. Long short-term memory model – A deep learning approach for medical data with irregularity in cancer predication with tumor markers. Comput Biol Med May 2022;144:105362.
[8] Sen S, Raghunathan A. Approximate computing for long short term memory (LSTM) neural networks. IEEE Trans Comput Aided Des Integr Circuits Syst Nov. 2018;37:2266–76.
[9] Liu X, Gherbi A, Li W, Cheriet M. Multi features and multi-time steps LSTM based methodology for bike sharing availability prediction. Procedia Comput Sci 2019; 155:394–401.
[10] Oruh J, Viriri S, Adegun A. Long short-term memory recurrent neural network for automatic speech recognition. IEEE Access 2022;10:30069–79.
[11] Hu J, Zheng W, "Transformation-gated LSTM: efficient capture of short-term mutation dependencies for multivariate time series prediction tasks", arXiv: 1907.03152, July 2019.
[12] Gao S, Huang Y, Zhang S, Han J, Wang G, Zhang M, et al. Short-term runoff prediction with GRU and LSTM networks without requiring time step optimization during sample generation. J Hydrol 2020;589:125188.
[13] Li Q, Xu Y. VS-GRU: a variable sensitive gated recurrent neural network for multivariate time series with massive missing values. Appl Sci 2019;9:3041.
[14] Mou L, Zhao P, Xie H, Chen Y. T-LSTM: a long short-term memory neural network enhanced by temporal information for traffic flow prediction. IEEE Access 2019;7: 98053–60.
[15] Ibraheem MR. Lower limb analysis based on surface electromyography (semg) using different time-frequency representation techniques. Int J Adv Sci Eng Information Technol 2023;13(1):24–33.
[16] Bansal H, Chinagundi B, Rana PS, Kumar N. An ensemble machine learning technique for detection of abnormalities in knee movement sustainability. Sustainability Oct. 2022;14(20):13464. https://doi.org/10.3390/su142013464.
[17] Zhu M, Guan X, Li Z, Gao Y, Zou K, Gao X, Wang Z, Li H, Cai K, Prediction of knee trajectory based on surface electromyogram with independent component analysis combined with support vector regression, Int J Adv Robotic Systems, vol. 19, p. 172988062211196, July 2022.
[18] Wang C, Guo Z, Duan S, He B, Yuan Y, Wu X. A real-time stability control method through semg interface for lower extremity rehabilitation exoskeletons. Front Neurosci 2021;15:1–12.
[19] Liu Y, Li X, Zhu A, Zheng Z, Zhu H, Design and evaluation of a surface electromyography-controlled lightweight upper arm exoskeleton rehabilitation robot, Int J Adv Robotic Systems, vol. 18, p. 172988142110034, May 2021.
[20] Tsinganos P, Cornelis B, Cornelis J, Jansen B, Skodras A. Data augmentation of surface electromyography for hand gesture recognition. Sensors 2020;20:4892.
[21] Morbidoni C, Cucchiarelli A, Fioretti S, Nardo FD. A deep learning approach to emg-based classification of gait phases during level ground walking. Electronics 2019;8:894.
[22] Khowailed IA, Abotabl A. Neural muscle activation detection: A deep learning approach using surface electromyography. J Biomech 2019;95:109322.
[23] Akhundov R, Saxby DJ, Edwards S, Snodgrass S, Clausen P, Diamond LE, Development of a deep neural network for automated electromyographic pattern classification, J Exp Biol, January 2019.
[24] Liu G, Zhang L, Han B, Zhang T, Wang Z, Wei P, "sEMG-Based Continuous Estimation of Knee Joint Angle Using Deep Learning with Convolutional Neural Network", IEEE 15th International Conference on Automation Science and Engineering (CASE), Vancouver, BC, Canada, August 2019.
[25] Benzaquen J, Shadmand MB, Mirafzal B. Ultrafast rectifier for variable-frequency applications. IEEE Access 2019;7:9903–11.
[26] Roy S, Azad ANMW, Baidya S, Khan F. A comprehensive review on rectifiers, linear regulators, and switched-mode power processing techniques for biomedical sensors and implants utilizing in-body energy harvesting and external power delivery. IEEE Trans Power Electron Nov. 2021;36:12721–45.
[27] Taylor L, Nitschke G, "Improving Deep Learning with Generic Data Augmentation", IEEE Symposium Series on Computational Intelligence (SSCI), Bangalore, India, pp. 1542-1547, Nov. 2018.
[28] Zanini RA, Colombini EL. Parkinson's disease EMG data augmentation and simulation with DCGANs and style transfer. Sensors (Basel) 2020;20:1–14.
[29] Khosla C, Saini BS, Enhancing Performance of Deep Learning Models with different Data Augmentation Techniques: A Survey, International Conference on Intelligent Engineering and Management (ICIEM), London, UK, Jun. 2020.
[30] Ibrahim M, Wedyan M, Alturki R, Khan MA, Al-Jumaily A. Augmentation in healthcare: augmented biosignal using deep learning and tensor representation. J Healthcare Eng 2021;2021:1–9.
[31] Nanni L, Paci M, Brahnam S, Lumini A. Comparison of different image data augmentation approaches. J Imaging 2021;7:1–21.
[32] Johannesen NJ, Kolhe ML, Goodwin M, "Comparing Recurrent Neural Networks using Principal Component Analysis for Electrical Load Predictions", 6th International Conference on Smart and Sustainable Technologies (SpliTech), Bol and Split, Croatia, Sep. 2021.
[33] Tao F, Liu G, "Advanced LSTM: A Study About Better Time Dependency Modeling in Emotion Recognition", IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Calgary, AB, Canada, vol. 6, pp. 19894-19901, Apr. 2018.
[34] https://mldta.com/dataset/emg-dataset-in-lower-limb/.
[35] Cao W, Yuan J, He Z, Zhang Z, He Z. Fast deep neural networks with knowledge guided training and predicted regions of interests for real-time video object detection. IEEE Access 2018;6:8990–9.
[36] Ukwandu O, Hindy H, Ukwandu E. An evaluation of lightweight deep learning techniques in medical imaging for high precision COVID-19 diagnostics. Healthcare Analytics Nov. 2022;2:1–9.

[37] Rorabaugh AK, Caíno-Lores S, Johnston T, Taufer M, High frequency accuracy and loss data of random neural networks trained on image datasets, Data in Brief, vol. 40, pp. 107780, Feb. 2022.
[38] Szepesi P, Szilágyi L. Detection of pneumonia using convolutional neural networks and deep learning. Biocybernetics Biomed Eng 2022;42:1012–22.
[39] Zhou Y, Wang X, Zhang M, Zhu J, Zheng R, Wu Q. MPCE: a maximum probability based cross entropy loss function for neural network classification. IEEE Access 2019;7:146331–41.
[40] Huang R, Huang C, Wang Y, Liu X. A deep learning-based algorithm for fetal ECG signal processing. IEEE Trans Biomed Eng 2021;68(4):1284–94.

Mai Ramadan Ibraheem, Assistant Prof. and Coordinator of IT dept., Faculty of Computers and Information, Kafrelsheikh University. Received Master of Science 2010, Mansoura University, in evaluating the network performance metrics and deciding the optimal routes using Artificial Intelligence techniques. Obtained Ph.D. 2014, Mansoura University, in identification and classification of various stages of focal liver lesions. Most of publications are in artificial intelligence, machine learning, computer vision, medical data analysis, and their applications. Interested in the field of machine learning using image and signal processing for medical diagnosis, have experience in this field and practiced novel techniques in the diagnosis of focal liver lesions, skin cancer lesion images and ocular diseases in retinal images. In addition to working on timing and classification of patello-femoral osteoarthritis patients based on sEMG that can assist neurophysiological standardization, evaluation and habitational detection. This work investigates function mapping from muscle activation as a real-time space to a PF osteoarthritis discriminative feature space. A member of IAENG since 2010. Current research interests are artificial intelligence, computer vision, medical image analysis, machine learning, pattern recognition, and biomedical engineering. She received a Freshman Academic Scholarship and graduated in the top 10% of her Master's and Ph.D. research cohorts at Mansoura University in Egypt. She currently works as the Executive Manager of the E-Learning Center at Kafrelsheikh University. She participated in refereeing at the International Space Apps Challenge 2020 and 2021, in addition to training for digital transformation as a certified trainer.

Saleh Naif Almuayqil : received the Ph.D. degree in computer science from the Faculty of Computing and Digital Technologies, Staffordshire University, U.K. Since 2018, he has been the Chair of the Information Systems Department. He is currently an Assistant Professor with the College of Computer and Information Sciences, Jouf University, Saudi Arabia. He has published scientific articles in various national and international journals and conferences. His current interests include health informatics, knowledge discovery, knowledge management, digital transformation, and data science.

A. A. Abd El-Aziz have completed Ph.D. degree in June 2014 in Information Science & Technology department from Anna University, Chennai-25, India. He has received B.Sc., and Master of Computer **Science** in 1999 and 2006 respectively from Faculty of Science, Cairo University. Now, he is an Associate Professor in the FGSSR, Cairo University, Egypt. He has 21 years' experience in teaching at Cairo University, Egypt. His research interests are database system, database security, Cloud computing, Big data, ML, DL and XML security. He has authored/coauthored over 72 research publications in peer-reviewed reputed journals and conference proceedings. He advised more than 17 master's graduates. Moreover, he has also served as a technical program committee member for many workshops and conferences and he has served as a reviewer for various international journals.

Medhat A. Tawfeek received the B.Sc., M.Sc. and PhD in Computer Science from Menoufia University, Faculty of Computers and Information in 2005, 2010 and 2015 respectively. He is presently an assistant professor in the department of Computer Science, Faculty of Computers and Information, Menoufia University, Egypt. He also holds the same position in in the department of Computer Science, College of Computer and Information Sciences, Jouf University, KSA. His research interest includes cloud computing, internet of things, smart card security, machine learning, distributed system, and fault tolerance.

Fatma M. Talaat: Machine Learning Department, Faculty of Artificial Intelligence, Kafrelsheikh University, Kafrelsheikh, Egypt. She received a B. Sc. in Computers and Systems Engineering. She got the PhD degree in fog computing. She got the master degree in the area of the security in network based applications. She is currently an Assistant Professor in the Faculty of Artificial Intelligence, Kafrelsheikh University.