Computers in Biology and Medicine 178 (2024) 108812

Contents lists available at ScienceDirect

Computers in Biology and Medicine

journal homepage: www.elsevier.com/locate/compbiomed

Meta-heuristic optimization algorithms based feature selection for joint moment prediction of sit-to-stand movement using machine learning algorithms

Ekin Ekinci [a], Zeynep Garip [a], Kasim Serbest [b,*]

[a] Department of Computer Engineering, Faculty of Technology, Sakarya University of Applied Sciences, Sakarya, Turkey
[b] Department of Mechatronics Engineering, Faculty of Technology, Sakarya University of Applied Sciences, Sakarya, Turkey

ARTICLE INFO

Keywords:
Manta ray foraging optimization (MRFO)
Marine predators algorithm (MPA)
Equilibrium optimizer (EO)
Extra tree regressor
Inverse dynamic analysis
Lower extremity

ABSTRACT

The sit-to-stand (STS) movement is fundamental in daily activities, involving coordinated motion of the lower extremities and trunk, which leads to the generation of joint moments based on joint angles and limb properties. Traditional methods for determining joint moments often involve sensors or complex mathematical approaches, posing limitations in terms of movement restrictions or expertise requirements. Machine learning (ML) algorithms have emerged as promising tools for joint moment estimation, but the challenge lies in efficiently selecting relevant features from diverse datasets, especially in clinical research settings. This study aims to address this challenge by leveraging metaheuristic optimization algorithms to predict joint moments during STS using minimal input data. Motion analysis data from 20 participants with varied mass and inertia properties are utilized, and joint angles are computed alongside simulations of joint moments. Feature selection is performed using the Manta Ray Foraging Optimization (MRFO), Marine Predators Algorithm (MPA), and Equilibrium Optimizer (EO) algorithms. Subsequently, Decision Tree Regression (DTR), Random Forest Regression (RFR), Extra Tree Regression (ETR), and eXtreme Gradient Boosting Regression (XGBoost Regression) ML algorithms are deployed for joint moment prediction. The results reveal EO-ETR as the most effective algorithm for ankle, knee, and neck joint moment prediction, while MPA-ETR exhibits superior performance for hip joint prediction. This approach demonstrates potential for enhancing accuracy in joint moment estimation with minimal feature input, offering implications for biomechanical research and clinical applications.

1. Introduction

Sit-to-stand (STS) is one of the most fundamental activities of daily living (ADL). This activity, repeated frequently throughout the day, involves coordinated movement of the lower and upper extremities [1]. The muscles of the lower extremities (ankle, knee, hip) facilitate body mobility, while the muscles of the upper extremities contribute to trunk stability. Contraction of the muscles around the ankle, knee, and hip generates joint moments that drive the STS movement [2,3]. Joint moments occurring during STS are crucial for various applications, such as identifying movement disorders in children [4], assessing balance disorders in adults [5], examining the impact of weight gain on joints [6], designing prosthetics and orthotics [7], and planning rehabilitation programs [8].

Classical mechanics-based mathematical methods are commonly employed for calculating joint moments in human limbs [9]. Furthermore, computer simulations aid in determining both joint moments and the muscle forces associated with these moments [10,11]. Specialized software packages like OpenSim [12] and AnyBody [13] are also utilized for joint moment analysis. These applications employ principles from classical mechanics to analyze the dynamics of joints and muscles. Given the complexity involved, various methods including mathematical techniques, computer simulations, and specialized software are necessary for accurately determining joint moments.

Recently, there has been increasing interest in employing artificial intelligence (AI) approaches to estimate joint moments. Machine learning (ML) algorithms, which have demonstrated successful applications across various domains [14–17], have also proven effective in accurately predicting joint moments and muscle forces during movements such as sit-to-stand (STS), walking, running, and sports exercises.

* Corresponding author.
E-mail addresses: ekinekinci@subu.edu.tr (E. Ekinci), zbatik@subu.edu.tr (Z. Garip), kserbest@subu.edu.tr (K. Serbest).

https://doi.org/10.1016/j.compbiomed.2024.108812
Received 15 March 2024; Received in revised form 19 June 2024; Accepted 24 June 2024
Available online 28 June 2024

Table 1
A summary of the previous ML studies on joint moment estimation. ANN; artificial neural network, DL; deep learning, EMG; electromyography, FNN; feed-forward neural network, IMU; Inertial Measurement Unit, LSTM; Long Short-Term Memory, WNN; Wavelet neural network.

Ref.	Movement	Algorithm	Moment calculation	Number of inputs/ features
[18]	STS	LSTM	Computer simulation	7–19
[21]	STS	ANN	Computer simulation	22
[22]	Treadmill walking, stair ascent/descent	DL	IMU sensor	102
[19]	Walking and ankle isokinetic dorsi/ plantar flexion	ANN	OpenSim, EMG	4
[23]	Treadmill walking	FNN	Rigid body dynamics	30
[24]	Treadmill running	ANN, Elastic net regression	OpenSim, EMG	14
[25]	Walking	WNN	OpenSim, EMG	18

Several studies have explored machine learning (ML)-based estimation of joint moments, utilizing diverse input data or features for training these algorithms. Importantly, incorporating a large amount of input data enhances the predictive accuracy of these algorithms [18]. However, setting up experiments with a substantial number of inputs is labor-intensive, posing a significant drawback for clinical studies across different disciplines. Electromyography (EMG) data is commonly used in training ML algorithms. For instance, Zhang et al. [19] successfully predicted ankle moments using an artificial neural network (ANN) model with four input data points derived from EMG measurements of three lower extremity muscles. EMG measurements involve intricate processes such as electrode placement, managing individual differences, and precise calibration [20]. Other studies, as listed in Table 1, have employed prediction models integrating multiple inputs including sensor data, motion analysis data, and ground reaction forces. Joint moments are of considerable interest to interdisciplinary researchers. Therefore, developing ML algorithms that accurately predict joint moments while minimizing the number of required input variables and simplifying experimental setups is crucial. The primary goal is to identify algorithms capable of achieving high prediction accuracy using a limited set of input data sources. However, a significant challenge in standard machine learning (ML) is the curse of dimensionality. There is a pressing need to develop algorithms that require simpler experimental setups and utilize fewer input data points for accurately estimating joint moments during human movements.

In the literature, meta-heuristic optimization algorithms have proven to be successful in overcoming the curse of dimensionality [26–28]. These algorithms typically start with a set of randomly generated solutions and use various search agents to identify the best solution. Extensive research has shown that meta-heuristic algorithms can effectively integrate with feature selection techniques. However, it's important to note that no single optimization algorithm can universally outperform all others in every optimization problem. Therefore, ongoing research continues to explore and improve feature selection methodologies.

In this study, we selected three meta-heuristic algorithms: MRFO, MPA, and EO, known for their effectiveness in solving optimization problems. The MRFO algorithm is noted for its strong merit-seeking capabilities and rapid convergence [29]. EO excels in both exploration and exploitation of solutions, demonstrating excellent convergence speed even for high-dimensional problems [30]. MPA, on the other hand, is appreciated for its simplicity, adjustable parameters, and flexible implementation [26]. Given these positive attributes, we chose to use MRFO, MPA, and EO in this study to address the feature selection problem effectively.

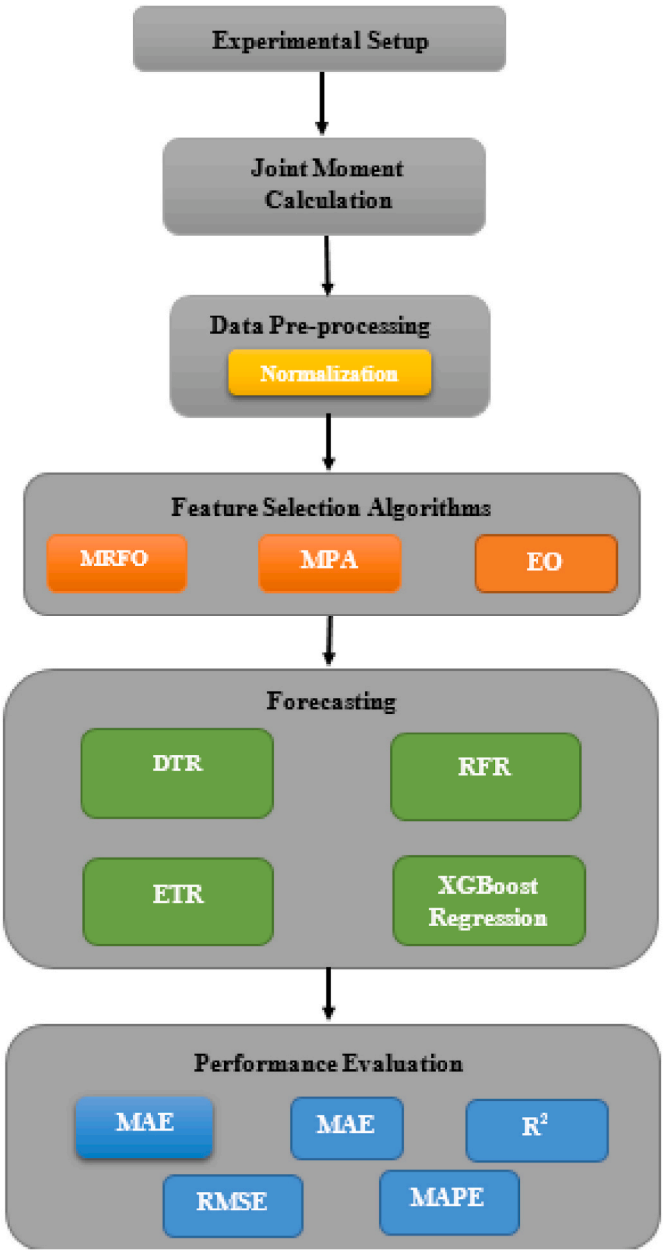

Fig. 1. Proposed architecture.

The aim of this study is to estimate the joint moments occurring in the lower extremity and neck joints during STS movement using Decision Tree Regression (DTR), Random Forest Regression (RFR), Extra Tree Regression (ETR), and eXtreme Gradient Boosting Regression (XGBoost Regression) ML algorithms with high accuracy and minimum number of features. The original dataset based on motion analysis and body segment inertial parameters was examined with Manta Ray Foraging Optimization (MRFO), Marine Predators Algorithm (MPA), and Equilibrium Optimizer (EO) optimization algorithms and the number of input data was reduced. Thus, joint moments were estimated with high forecasting accuracy using a small number of input data. The models were evaluated in terms of Mean Absolute Error (MAE), Mean Square Error (MSE), Root Mean Square Error (RMSE), correlation coefficient (R^2), and Mean Absolute Percentage Error (MAPE). The contributions of our study to the literature can be summarized as follows:

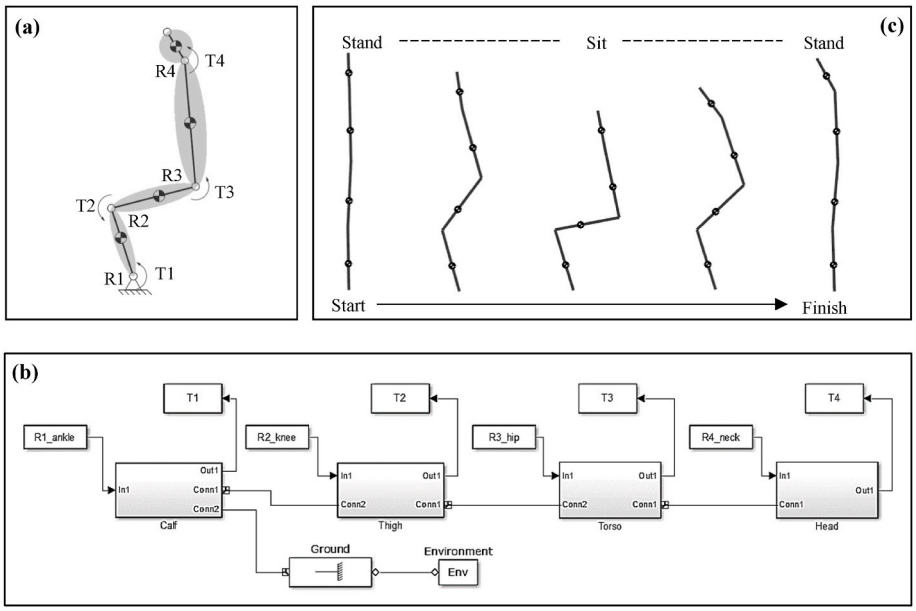

Fig. 2. The musculoskeletal model of the STS. **a**; A four DOF mechanical model, **b**; MATLAB Multibody diagram of the STS, **c**; Simulation views. 1; ankle, 2; knee, 3; hip, 4; neck; R; revolute joint, T; joint torque, [21].

- Meta-heuristic optimization algorithms were employed to achieve highly accurate estimates of joint moments using a limited amount of input data.
- The study successfully estimated the neck joint moment, which exhibits a distinct motion pattern compared to the ankle, knee, and hip joints.
- The study achieved highly accurate estimation of joint moments for participants across various BMI categories.

In the remainder of the paper, the methods section will detail the dataset used, feature selection utilizing meta-heuristic optimization algorithms, and the prediction of joint moments using machine learning (ML) algorithms. The results section will present the performance evaluation of these ML algorithms and discuss the outcomes regarding prediction success. In the discussion section, comparisons with previous studies will be made, the findings of our study will be critically evaluated, and potential applications will be discussed. Finally, the conclusion section will provide a comprehensive summary, evaluate the overall findings, and emphasize potential avenues for future research.

2. Materials and methods

The proposed system architecture for predicting joint moments using sit-to-stand (STS) motion and biomechanical data through feature selection is illustrated in Fig. 1. Initially, experimental data undergoes analysis and normalization. Subsequently, meta-heuristic optimization algorithms are employed for feature selection to identify relevant features. Finally, the selected features are inputted into regression algorithms. The primary objective of the proposed system is to develop an effective and precise feature selection method that ensures the accuracy of joint moment prediction.

2.1. Preparation of data set

The dataset utilized in this article is adapted from Cilli et al. (2021) [31]. It comprises motion analysis results from STS movements, calculations of body segment inertial parameters, and joint moments derived from MATLAB Multibody simulations.

2.1.1. Experimental setup of the STS

To determine the angular displacement of the joints during STS movements, a motion analysis was conducted involving 20 participants (5F/15 M) categorized into four different body mass index (BMI) groups (underweight, normal, overweight, and obese). This approach ensured the dataset captured a range of physical characteristics relevant to the movement. During the movement analysis, passive markers were placed on anatomical landmarks including the head, shoulders, hips, knees, ankles, and feet of the subjects. Participants initiated the STS motion

Table 2
A sample of dataset for ankle joint.

Sub. No.	Inputs							Outputs
	Time (s)	Angle (deg.)	Length (cm)	Mass (kg)	Moment of inertia (kg·m^2)			Joint moment (Nm)
					I_{xx}	I_{yy}	I_{zz}	
1	1.023	115.7	8.8	2.4553	0.045554	0.045554	0.003902	**3.31876**
2	1.056	124	7.2	3.6292	0.110263	0.110263	0.008004	**−9.81424**
2	1.089	124.3	7.2	3.6292	0.110263	0.110263	0.008004	**−11.36865**
3	1.122	113.4	7.5	3.4142	0.059902	0.059902	0.005779	**−23.22504**
3	1.155	112.7	7.5	3.4142	0.059902	0.059902	0.005779	**−20.26088**
3	1.188	112	7.5	3.4142	0.059902	0.059902	0.005779	**−17.82811**
4	1.221	121.6	8.4	3.3067	0.057939	0.057939	0.006651	**11.57902**
4	1.254	121.3	8.4	3.3067	0.057939	0.057939	0.006651	**12.84966**
4	1.287	120.6	8.4	3.3067	0.057939	0.057939	0.006651	**13.04558**
4	1.32	119.9	8.4	3.3067	0.057939	0.057939	0.006651	**12.44536**

from a standing position, sat on a chair without back support, and then returned to the initial standing position. Throughout this process, digital camera footage (30 fps) captured the movements in the sagittal plane. The camera images were subsequently analyzed using Tracker [32], a video analysis and modeling tool software. This allowed for the calculation of angular displacements for the neck, hip, knee, and ankle joints based on the motion of the markers.

2.1.2. Calculation of the body inertial parameters and joint moments

Inertial parameters of the limbs such as length, mass, moment of inertia (I_{xx}, I_{yy}, I_{zz}), center of mass were determined by measurements and calculations according to the Yeadon protocol [33]. A link-segment model was constructed using MATLAB Multibody tools to calculate joint moments during the STS movement. This model includes segments representing the calf, thigh, torso, and head, along with joints for the neck, hip, knee, and ankle. The link-segment model is structured as an open chain mechanical system with a total of 4 limbs and 4 degrees of freedom (DOF). Given the symmetrical movement of the entire body during STS, the link-segment model was designed to represent half of the body. Fig. 2 illustrates the link-segment model and MATLAB block diagrams used in the analysis.

2.1.3. Structure of the data

The dataset encompasses diverse measurements including motion analysis, body inertial parameters, and joint moments. Table 2 illustrates sample measurements specifically for the ankle joint. Similar data formats are applied to the knee, hip, and neck joints. The dataset maintains consistency across subjects and time intervals. For each joint, the variables recorded include time, angle, and joint moment. Meanwhile, parameters such as length, mass, and moment of inertia remain constant throughout the dataset. This structured approach ensures uniformity and facilitates comparative analysis across different joints and participants. For each joint there are seven input data (time, angle, length, mass, I_{xx}, I_{yy}, I_{zz}) and one output data (joint moment). Taking all joints into account, the complete dataset contains 25 inputs and 4 outputs respectively. Positive or negative values assigned to the joint moment indicate its direction relative to the joint.

The accuracy of the models is adversely affected by imbalance in the interquartile range. A crucial pre-processing step for a dataset with a significant interquartile range and distribution is normalization, especially when utilizing different units for quantification. Among normalization methods, in this study, min-max normalization is used to map the range to [0, 1]. Accurate forecasting is made possible by min-max normalization, which preserves the relation between data instances. Min-max normalization is calculated as follows:

$$x_{i\_normalized=}\frac{x_i-x_{min}}{x_{max}-x_{min}} \tag{1}$$

where x_i represents the i_{th} data instance, x_{min} represents the minimum-valued data instance and x_{max} represents the maximum-valued data instance. $x_{i\_normalized}$ is in the range of $[0, 1]$ and the normalized form of x_i.

2.2. Meta-heuristic algorithms

2.2.1. Manta ray foraging optimization (MRFO)

Zhao et al. developed the MRFO algorithm inspired by the behavior of the mantle ray [34]. The MRFO algorithm is based on the foraging behavior of mantas, including chain foraging, cyclone foraging and somersault foraging [35]. These foraging methods are used to update solutions during optimization.

The following is a description of the mathematical forms of the processes:

Chain foraging: The manta ray assesses the location of the plankton during the aquatic chain search behavior. The manta ray moves towards the better positioned plankton. In other words, the plankton with the highest density provides the best solution. Chain foraging method is modeled by equation (1). In Equation (2), x_i^d is the position of the manta ray, r is a random value in the range $[0, 1]$, α calculated with $2.r.\sqrt{|\log(r)|}$ is the weight coefficient, and $x_{best}^d(t)$ is the highest represents plankton with density [30].

$$x_i^d(t+1)=\begin{cases} x_i^d(t)+r.\left(x_{best}^d(t)-x_i^d(t)\right)+\alpha.\left(x_{best}^d(t)-x_i^d(t)\right) & i=1 \\ x_i^d(t)+r.\left(x_{i-1}^d(t)-x_i^d(t)\right)+\alpha.\left(x_{best}^d(t)-x_i^d(t)\right) & i=2,\ldots,N \end{cases} \tag{2}$$

Cyclone foraging: The manta rays create a lengthy chain during this phase of foraging, and each particle follows a spiral path. The best particle and its front particle are used to update the particle's position. Cyclone foraging method is modeled by Equation (2). In Equation (3), r_1 represents a random value in the range $[0, 1]$, β, calculated with $2e^{r_1\frac{T-t+1}{T}}\sin(2\pi r_1)$ represents the weight coefficient, and T represents the maximum number of iterations [34].

$$x_i^d(t+1)=\begin{cases} x_{best}^d(t)+r.\left(x_{best}^d(t)-x_i^d(t)\right)+\beta.\left(x_{best}^d(t)-x_i^d(t)\right) & i=1 \\ x_{best}^d(t)+r.\left(x_{i-1}^d(t)-x_i^d(t)\right)+\beta.\left(x_{best}^d(t)-x_i^d(t)\right) & i=2,\ldots,N \end{cases} \tag{3}$$

To enhance exploration, the spinning search behavior is implemented. As a result, a new position is given in the search space that influences the exploration activity. This causes a thorough global search, which is shown in Equation (4). In Equation (5), x_{rand}^d gives the random position relative to Lb^d lower bound and Ub^d upper bound.

$$x_i^d(t+1)=\begin{cases} x_{rand}^d(t)+r.\left(x_{rand}^d(t)-x_i^d(t)\right)+\beta.\left(x_{rand}^d(t)-x_i^d(t)\right) & i=1 \\ x_{rand}^d(t)+r.\left(x_{i-1}^d(t)-x_i^d(t)\right)+\beta.\left(x_{rand}^d(t)-x_i^d(t)\right) & i=2,\ldots,N \end{cases} \tag{4}$$

$$x_{rand}^d=Lb^d+r.\left(Lb^d-Ub^d\right) \tag{5}$$

Somersault foraging: The exploitation phase is modeled by the somersault foraging technique. During this stage, Equation (6) is used to update the particle positions as the particles swim to and around the meal. In the equation, r_2 and r_3 represent random numbers in the range $[0,1]$, and S represents somersault factor (equal to 2) [35,36].

$$x_i^d(t+1)=x_i^d(t)+S.\left(r_2x_{best}^d(t)-r_3x_i^d(t)\right), i=1,\ldots.,N \tag{6}$$

2.2.2. Marine Predators Algorithm (MPA)

MPA was conceived by Faramarzi et al., drawing inspiration from the dynamic interplay observed in the predator-prey social dynamics of marine ecosystems [37]. MPA is a heuristic optimization algorithm crafted around the encounter rate observed between marine predators and their prey [38]. The initial solution of MPA begins with a randomized distribution across the search space. Building upon MPA, the algorithm's phase transitions are governed by the speed ratio between the prey and the predator [39]. Marine predators undergo three phases to complete their step size while capturing their prey [40]. The primary characteristic of the initial phase of the algorithm is its notable high speed, whereas in subsequent phases, emphasis is placed on unity and weaker ratios.

The starting solution is established through the random and uniformly distributed search space of MPA. The number of predators is n, the iterations are represented by m, the optimization parameter size is d, and *Prey* signifies the initial position of the prey. X_{max} and X_{min} in Equation (7) are the maximum and minimum values, and rand is a random vector in the range $[0,1]$.

$$X_0=X_{min}+rand(X_{max}-X_{min}) \tag{7}$$

In this section, the *Prey* matrix, containing the initial population's positions, combines with the *Elite* matrix showcasing the best fitness function. Each stage is outlined as follows.

Since *Prey* has high speed in Phase 1, the *Predator*'s movement must be stopped. This process is executed for only a third of the total iterations. *Prey*'s behavior is determined by Brownian motion. The matrices used by *Prey* are updated with Equation (9). $P = 0.5$ in Equations (8) and (9), R is defined as uniformly distributed random numbers between $[0, 1]$ and R_B is defined as a vector comprising random numbers generated from the normal distribution resembling Brownian motion.

$$\overrightarrow{step_i} = \overrightarrow{R_B} \otimes \left(\overrightarrow{Elite_i} - \left(\overrightarrow{R_B} \otimes \overrightarrow{Prey_i}\right)\right) \tag{8}$$

$$\overrightarrow{Prey_i} = \overrightarrow{Prey_i} + \left(P.\overrightarrow{R} \otimes \overrightarrow{step_i}\right) \tag{9}$$

During Phase 2, both *Prey* and *Predator* advance at equal speeds, encompassing the latter two-thirds of the algorithm. Here, they employ distinct movement strategies: *Predator* utilizes Brownian motion, while *Prey* employs Levy motion. Specifically, *Prey* undergoes multiplication by a vector constituted of random numbers generated from the normal distribution of Levy's motion (R_L). The movements of the initial half of the population are then updated in accordance with Equations (10) and (11).

$$\overrightarrow{step_i} = \overrightarrow{R_L} \otimes \left(\overrightarrow{Elite_i} - \left(\overrightarrow{R_L} \otimes \overrightarrow{Prey_i}\right)\right) \tag{10}$$

$$\overrightarrow{Prey_i} = \overrightarrow{Prey_i} + \left(P.\overrightarrow{R} \otimes \overrightarrow{step_i}\right) \tag{11}$$

The remaining half of the population is updated based on Equations (12) and (13). The *Elite* matrix undergoes multiplication by R_B. Here, *CF* serves as an adaptive parameter regulating the step size for *Predator* movement.

$$\overrightarrow{step_i} = \overrightarrow{R_B} \otimes \left(\left(\overrightarrow{R_B} \otimes \overrightarrow{Elite_i}\right)\right) \tag{12}$$

$$\overrightarrow{Prey_i} = \overrightarrow{Elit_i} + \left(P.\ CF \otimes \overrightarrow{step_i}\right) \tag{13}$$

$$CF = \left[1 - \left(Iter./Max.Iter\right)\right]^{\left(2.Iter./Max.Iter.\right)} \tag{14}$$

During Phase 3, the *Prey* is presumed to move at a slower pace compared to the *Predator*, employing Levy movement exclusively for the remainder of the iteration. At this juncture, the *Elite* matrix undergoes multiplication by R_L. The *Prey* matrix is updated utilizing Equation (16).

$$\overrightarrow{step_i} = \overrightarrow{R_L} \otimes \left(\left(\overrightarrow{R_L} \otimes \overrightarrow{Elite_i}\right) - \overrightarrow{Prey_i}\right) \tag{15}$$

$$\overrightarrow{Prey_i} = \overrightarrow{Prey_i} + \left(P.CF \otimes \overrightarrow{Prey_i}\right) \tag{16}$$

In MPA, following each iteration, the *Elite* matrix is substituted with the best solutions. Furthermore, the solution achieved upon reaching the maximum number of iterations or meeting the algorithm's stopping criterion is regarded as the final solution.

2.2.3. Equilibrium Optimizer (EO)

It is a physics-based algorithm created by Faramarzi et al. by modeling the dynamic mass balance mechanism operating on the EO control volume [41]. According to the method, each particle represents the concentration in the control volume at a particular instant, and as time goes on, the concentration converges toward the concentration at equilibrium [42].

The mathematical forms of the processes are described below:

Initialization: In the EO algorithm, the values of individuals in the optimization process search space are determined according to the equation given in Equation (17). Throughout the process, every particle updates its focus based on the best solutions, referred to as equilibrium candidates. $C_i^{initial}$ is the initial concentration vector value of the particle, C_{min} and C_{max} are the minimum and maximum values for the dimensions, n is the particle number of the population, and *rand* is random numbers in the range $[0, 1]$.

$$C_i^{initial} = C_{min} + rand_i(C_{max} - C_{min}) \quad i = 1, 2, \ldots, n \tag{17}$$

Equilibrium pool and candidates: The objective of the EO algorithm is to seek equilibrium within the system during its operation. Therefore, the best four particles and their concentrations are selected from the candidate solutions in the algorithm. The fifth equilibrium particle is derived by calculating the arithmetic mean of the chosen particles. The vectors of the balance pool are created using Equation (18).

$$\begin{aligned} \overrightarrow{C}_{eq.pool} &= \left\{\overrightarrow{C}_{eq(1)}, \overrightarrow{C}_{eq(2)}, \overrightarrow{C}_{eq(3)}, \overrightarrow{C}_{eq(4)}, \overrightarrow{C}_{eq(ave)}\right\} \\ \overrightarrow{C}_{(ave)} &= \frac{\overrightarrow{C}_{eq(1)} + \overrightarrow{C}_{eq(2)} + \overrightarrow{C}_{eq(3)} + \overrightarrow{C}_{eq(4)}}{4} \end{aligned} \tag{18}$$

Update candidate solutions: The EO algorithm comprises two phases: exploration and exploitation. During the exploration phase, the algorithm utilizes the top four candidate solution values. The exploitation phase's effectiveness is bolstered by the average value derived from these four computed candidates.

The equilibrium between these two phases, as outlined in Equation (19), is attained through the incorporation of an exponential factor, F. It is a random vector between λ and $[0, 1]$, the t value decreases as the number of iterations increases. *iter* and max_{iter} depict the present iteration count and the maximum iteration limit, respectively, and a_2 represents the exploration ability constant. In this context, as the value of a_2 increases, the exploration capability becomes more pronounced, consequently leading to a decrease in exploitation capability. At value a_1, a_2 is the opposite of the variable's situation. The larger the value of a_1, the higher its impact on the exploitation phase. On the contrary, in the discovery phase, its effect is less.

$$\begin{aligned} F &= a_1 sign(r - 0.5)\left[e^{-\lambda t} - 1\right] \\ t &= \left(1 - \frac{iter}{\max\_iter}\right)^{\left(a_2 \frac{iter}{\max\_iter}\right)} \end{aligned} \tag{19}$$

The impact of the production rate (G) plays a significant role in the exploitation stage of the EO algorithm. This rate is described by Equation (20). G_0 represents the initial value, r_1 and r_2 a number that varies randomly between $[0, 1]$, *GCP* denotes the control parameter governing the production rate, while *GP* signifies the actual production rate utilized in the status update.

$$\begin{aligned} G &= G_0 F \\ \overrightarrow{G_0} &= \overrightarrow{GCP}\left(\overrightarrow{G_{eq}}\right) - \lambda C \\ \overrightarrow{GCP} &= \begin{cases} 0.5r_1 & r_2 \geq GP \\ 0 & r_2 \leq GP \end{cases} \end{aligned} \tag{20}$$

Finally, the conditions of the particles are described using the comprehensive update principle outlined in Equation (21). C and C_{eq} refer to the present particle and the equilibrium candidate solution.

$$C = C_{eq} + \left(C - C_{eq}\right)F + \frac{G}{\lambda V}(1 - F) \tag{21}$$

2.3. Classification algorithms

2.3.1. Decision Tree Regression (DTR)

Decision trees (DTs) are useful for understanding the relationship between input features and the target variable because they are simple to understand and visualize. They can capture complex nonlinear interactions between input variables and the target variable effectively. The data from the training set is recursively partitioned, and several splits are performed on the data to create a DT model [43]. The

algorithm starts at the root node of the tree and proceeds down the tree by following splitting decisions based on the input feature values until it reaches a leaf node to make predictions for new inputs. The prediction is then made using the numerical value assigned to that leaf node. Recursive splitting continues until a stopping criterion is met, such as limiting the depth of the tree, having a minimum number of data points at a leaf node, or reaching a certain number of nodes. When a stopping criterion is met, the algorithm assigns a numerical value to the leaf node based on the average (or another statistic like median) of the target variable's data points in that node. This assigned value serves as the prediction for any additional data points that enter that leaf node. The algorithm first selects the dataset characteristic that best divides the data into two subsets. The splitting decision is based on minimizing the variance of the target variable within each subgroup.

2.3.2. Random forest regression (RFR)

Due to its superior performance, simplicity of use, and low computing cost, random forest regression (RFR) has gained a lot of popularity in recent years. It is a Breiman-created ensemble learning method that is based on the combination of several regression trees [44]. Every decision tree in the random forest regression (RFR) ensemble is trained using a different random subset of the training data, which helps mitigate overfitting [45]. A random sample (with replacement) is drawn from the training set for each tree in the ensemble, a process known as bootstrapping. Consequently, each tree is trained on a slightly different subset of the data. The training process involves a series of threshold tests where randomly selected features are compared against randomly chosen thresholds to partition the data into two branches at each node. This iterative splitting continues until the leaf nodes, or terminal nodes, of the tree contain just one or a predetermined number of data points. Each training data point at the leaf nodes has a corresponding output value, which are aggregated to determine the final value assigned to each leaf. Once a tree is constructed, a test data point follows a specific path through the tree based on the threshold tests until it reaches a leaf node that outputs the predicted value corresponding to that data point. The remaining training set, not included in the bootstrap samples for a given tree, can serve as a validation set. The tree's performance on this validation set is evaluated using regression model metrics to determine its effectiveness. The training process continues until the forest reaches the predetermined number of regression trees specified by the user. The ensemble of regression values produced by the forest can then be used to calculate the final regression value, typically by averaging (or occasionally using the median) of the individual tree predictions.

2.3.3. Extra Tree Regression (ETR)

ETR devised by Geurts et al. [46], is an ensemble learning model built on extensive decision trees and originally derived from the Random Forest (RF). Unlike RF, in ETR, nodes of decision tree are split randomly into two with simpler and efficient way. Moreover, due to its high unpredictability, to reduce the variance and solves the issue of over-fitting ETR makes use of all the data points.

ETR algorithm constructs multiple decision trees similar to Random Forests but introduces additional randomness. Unlike Random Forests, Extra Trees randomly selects a subset of features and then chooses the best split from those features, rather than selecting the best split for each node in a tree. This added randomness helps reduce the model's variance.

Similar to Random Forests, Extra Trees also utilizes bootstrap sampling for each tree in the ensemble. This involves randomly selecting subsets of the training data with replacement to train each individual tree. This further enhances the diversity among the trees in the ensemble.

During inference, each tree in the ensemble provides a prediction. The final prediction for regression tasks is typically computed as the average of these individual tree predictions. This ensemble averaging technique helps mitigate overfitting and improves the model's ability to generalize to new data.

2.3.4. eXtreme gradient boosting (XGBoost) regression

Based on the gradient boosting architecture, XGBoost generates weak learners, typically shallow decision trees, that focus on correcting the errors of prior models to build a robust predictive model [47]. This sequential learning process effectively produces an accurate and precise regression model, simultaneously reducing both bias and variance.

XGBoost enhances its loss function by integrating regularization terms with a regression loss (such as mean squared error for regression tasks). This approach penalizes complex models while rewarding those that generalize well, thereby guiding the learning process effectively.

Moreover, XGBoost provides a feature importance metric that helps identify the input characteristics most influential in making regression predictions. This metric is valuable for feature selection and gaining insights into how different variables impact the target variable.

To avoid overfitting, XGBoost uses regularization terms such L1 (Lasso) and L2 (Ridge) regularization. Using hyperparameters, regularization's strength of model can be regulated.

XGBoost is made with efficiency and speed in mind. It is appropriate for huge datasets because it can benefit from distributed computing and parallel processing.

3. Results and discussion

The primary goal of experimental study is to develop an efficient feature selection technique that accurately forecasts joint moments.

The experiments were performed on Google Colab with the following system setup: a GPU Tesla K80 featuring 12 GB of GDDR5 VRAM, alongside an Intel Xeon Processor equipped with two 2.20-GHz cores and 13 GB of RAM. For the MRFO, EO, and MPA algorithms, a population size of 30 and 100 iterations were selected. Regression algorithms were implemented using the PyCaret[1] library.

An important factor in forecasting the final value needed to implement the RFR, ETR and XGBoost regression models is the number of trees [48]. For the forecasting, a total of 100 trees were chosen for each of them.

3.1. Performance metrics

To see the effect of feature selection algorithms on the forecasting performance we use MAE, MSE, RMSE, R^2, MAPE. MAE is the average of the difference between actual and the predicted value of joint moments. While MSE is the average of the square of the difference between actual and the predicted value of joint moments, RMSE is square root of MSE. R^2 assesses how close the actual values of joint moments are to the fitted regression line. MAPE is the average of the absolute percentage errors of the actual and the predicted value of joint moments.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{22}$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{23}$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{24}$$

$$R^2 = 1 - \sum_{i=1}^{n}\frac{(y_i - \hat{y}_i)^2}{(y_i - \bar{y})^2} \tag{25}$$

[1] https://pycaret.org/.

Table 3
Outcomes of meta-heuristic algorithms applied to datasets.

Dataset	Algorithm	Min	Mean	Std	Selected Features
Ankle	MRFO	0.033	0.045	0.008	Time, Knee Mass, Hip Ixx
	MPA	0.025	0.043	0.008	Time, Neck Height
	EO	0.028	0.043	0.010	Time, Foot Height, Foot Iyy, Hip Izz
Knee	MRFO	0.015	0.027	0.005	Time, Foot Height, Hip Iyy
	MPA	0.015	0.027	0.005	Time, Neck Height
	EO	0.013	0.025	0.005	Time, Knee Height, Hip Izz
Hip	MRFO	0.019	0.034	0.008	Time, Knee Height
	MPA	0.022	0.033	0.007	Time, Foot Height, Hip Iyy
	EO	0.015	0.033	0.007	Time, Knee Height
Neck	MRFO	0.029	0.041	0.006	Time, Knee Height
	MPA	0.024	0.043	0.010	Time, Foot Height, Foot Iyy, Hip Ixx
	EO	0.024	0.040	0.008	Time, Foot Height, Foot Ixx, Hip Iyy

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left(|y_i - \hat{y}_i| / y_i\right) * 100 \quad (26)$$

In the equations above, the expected value, the predicted value, and the mean of the joint moments are represented by y_i, $\hat{y}_i$ and $\bar{y}$, respectively; n is the number of observations in the test set. R^2 is anticipated to be high, whereas MAE, MSE, MAPE, and RMSE are anticipated to be low. Our findings indicate that the MAE is the most accurate means of determining average error magnitude.

3.2. Evaluation of the results

The experimental results, as depicted in Table 3 shows the minimum (min), mean and standard deviation (std) values obtained by applying metaheuristic algorithms to datasets and alongside the features selected based on the minimum criterion.

The raw dataset consists of 25 features. Analysis of meta-heuristic algorithm outcomes, as presented in Table 3, indicates that the number of selected features ranges from a minimum of 2 to a maximum of 4. Since joint moments vary over time, it is expected that the time feature is included in all algorithms. Other selected features across algorithms include mass, height, and moment of inertia, which vary depending on the specific algorithm. Upon reviewing the fitness function values, it is clear that the EO emerges as the most effective algorithm for predicting knee, hip, and neck joint moments. Conversely, the MPA stands out as the top performer for predicting ankle joint moments. The choice of selected features, in addition to the fitness function, plays a crucial role in predicting joint moments. For ankle, knee, and neck joint moment predictions, the EO algorithm is preferred due to its high prediction success and its potential to create a more suitable experimental setup. Similarly, the MPA algorithm is recommended for hip joint moment estimation. The convergence speed of feature selection algorithms is compared in Fig. 3 for each dataset.

From Fig. 3, it can be concluded that while MPA converges much faster than MRFO and EO for ankle, knee and neck dataset, EO converges much faster than MRFO and MPA for hip dataset.

The objective of employing feature selection methods is to achieve model performance comparable to using all features while using a reduced set of features. To illustrate this, predictions were generated using the original feature set and the feature sets obtained after conducting feature selection. The results are presented in Table 4.

Using the evaluation metrics described in the previous section, the feature sets obtained with the recommended feature selection algorithms - MRFO, MPA, and EO - were compared against the original feature set. The best-performing regression models for each combination of algorithm and feature selection method are highlighted in Table 4. Across all datasets, ETR consistently outperforms DTR, RFR, and XGBoost Regression among the four regression models. The use of feature selection methods offers the advantage of achieving model performance comparable to using all features while employing fewer features. For instance, MRFO optimizes the performance of all four algorithms with 3 features in the ankle dataset, while MPA optimizes DTR and RFR with only 2 features in the knee dataset. In contrast, for the hip dataset, MPA optimizes all algorithms with 3 features, and for the neck dataset, both MPA and EO optimize performance using four features. Examining the findings presented in Table 4 reveals that optimizing the dataset with meta-heuristic algorithms enhances the accuracy of moment estimation across all joints.

For a more comprehensive comparison between the original feature set and the selected feature sets, Fig. 4 presents the average prediction errors of the best models for each dataset. The results indicate that models constructed using selected features exhibit better data fitting capabilities. Overall, when considering various datasets, feature

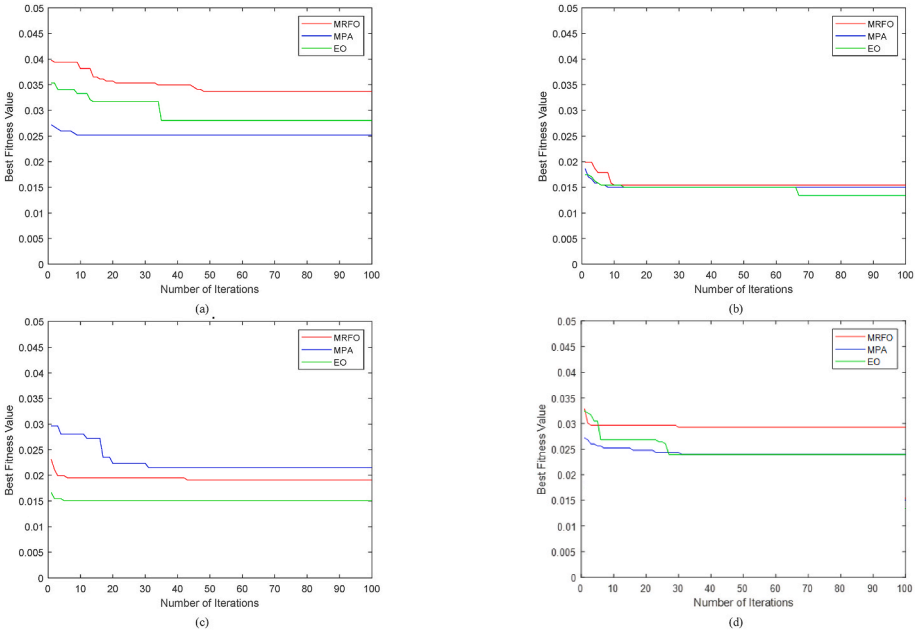

Fig. 3. Convergence graph for (a) ankle dataset (b) knee dataset (c) hip dataset (d) neck dataset.

Table 4
Performance evaluation for different feature set and 3 outputs (ankle, knee, hip and neck) with ML algorithms. DTR; Decision Tree Regression, RFR: Random Forest Regression, ETR; Extra Tree Regression, XGBoost Regression; eXtreme Gradient Boosting Regression, MRFO; Manta Ray Foraging Optimization, MPA; Marine Predators Algorithm, EO; Equilibrium Optimizer.

Dataset	Algorithm	Feature Set	MAE	MSE	RMSE	R^2	MAPE
Ankle	**DTR**	Original	0.0291	0.0029	0.0533	0.9095	0.0749
		MRFO	**0.0189**	0.0012	0.0338	0.9616	0.0471
		MPA	**0.0197**	0.0013	0.0362	0.9589	0.0489
		EO	**0.0188**	0.0012	0.0338	0.9636	0.0469
	RFR	Original	0.0212	0.0012	0.0336	0.9640	0.0646
		MRFO	**0.0167**	0.0008	0.0272	0.9765	0.0462
		MPA	**0.0187**	0.0010	0.0310	0.9695	0.0509
		EO	**0.0153**	0.0006	0.0248	0.9803	0.0427
	ETR	Original	0.0107	0.0004	0.0186	0.9890	0.0342
		MRFO	**0.0082**	0.0003	0.0159	0.9919	0.0258
		MPA	0.0279	0.0029	0.0532	0.9115	0.0781
		EO	**0.0055**	0.0001	0.0109	0.9961	0.0186
	XGBoost Regression	Original	0.0218	0.0012	0.0338	0.9637	0.0499
		MRFO	**0.0208**	0.0009	0.0296	0.9719	0.0713
		MPA	0.0226	0.0011	0.0328	0.9663	0.1078
		EO	**0.0194**	0.0008	0.0290	0.9737	0.0748
Knee	**DTR**	Original	0.0160	0.0009	0.0292	0.9697	0.0309
		MRFO	**0.0118**	0.0005	0.0218	0.9827	0.0216
		MPA	**0.0121**	0.0005	0.0209	0.9834	0.0218
		EO	**0.0112**	0.0004	0.0199	0.9848	0.0207
	RFR	Original	0.0124	0.0004	0.0198	0.9866	0.0258
		MRFO	**0.0089**	0.0002	0.0146	0.9927	0.0154
		MPA	**0.0112**	0.0003	0.0181	0.9885	0.0184
		EO	**0.0091**	0.0002	0.0153	0.9919	0.0157
	ETR	Original	0.0061	0.0001	0.0102	0.9964	0.0133
		MRFO	0.0064	0.0001	0.0118	0.9952	0.0107
		MPA	0.0183	0.0011	0.0329	0.9621	0.0296
		EO	**0.0054**	0.0001	0.0099	0.9965	0.0091
	XGBoost Regression	Original	0.0123	0.0004	0.0188	0.9878	0.0239
		MRFO	**0.0120**	0.0003	0.0176	0.9894	0.0253
		MPA	0.0127	0.0003	0.0184	0.9881	0.0268
		EO	**0.0123**	0.0004	0.0186	0.9879	0.0304
Hip	**DTR**	Original	0.0243	0.0019	0.0429	0.9417	0.0696
		MRFO	0.0376	0.0059	0.0767	0.8219	0.0878
		MPA	**0.0206**	0.0019	0.0409	0.9453	0.0570
		EO	0.0376	0.0059	0.0767	0.8219	0.0878
	RFR	Original	0.0184	0.0009	0.0302	0.9722	0.0701
		MRFO	0.0358	0.0044	0.0661	0.8675	0.0992
		MPA	**0.0161**	0.0007	0.0259	0.9796	0.0504
		EO	0.0358	0.0044	0.0661	0.8675	0.0992
	ETR	Original	0.0109	0.0003	0.0184	0.9898	0.0356
		MRFO	0.0285	0.0049	0.0692	0.8540	0.0754
		MPA	**0.0052**	0.0001	0.0094	0.9972	0.0193
		EO	0.0285	0.0049	0.0692	0.8540	0.0754
	XGBoost Regression	Original	0.0181	0.0008	0.0274	0.9765	0.0605
		MRFO	0.0386	0.0045	0.0668	0.8641	0.1070
		MPA	**0.0193**	0.0008	0.0284	0.9756	0.0629
		EO	0.0386	0.0045	0.0668	0.8641	0.1070
Neck	**DTR**	Original	0.0294	0.0038	0.0594	0.8951	0.0616
		MRFO	0.0384	0.0059	0.0756	0.8397	0.0736
		MPA	**0.0206**	0.0014	0.0370	0.9603	0.0445
		EO	**0.0219**	0.0023	0.0444	0.9384	0.0501
	RFR	Original	0.0220	0.0016	0.0389	0.9560	0.0501
		MRFO	0.0343	0.0039	0.0623	0.8926	0.0830
		MPA	**0.0163**	0.0008	0.0274	0.9789	0.0428
		EO	**0.0170**	0.0008	0.0282	0.9778	0.0461
	ETR	Original	0.0120	0.0005	0.0212	0.9875	0.0263
		MRFO	0.0264	0.0039	0.0619	0.8933	0.0516
		MPA	**0.0048**	0.0001	0.0087	0.9978	0.0111
		EO	**0.0048**	0.0001	0.0085	0.9979	0.0110
	XGBoost Regression	Original	0.0210	0.0012	0.0346	0.9668	0.0449
		MRFO	0.0365	0.0037	0.0606	0.8976	0.0750
		MPA	**0.0204**	0.0009	0.0306	0.9741	0.0473
		EO	**0.0208**	0.0010	0.0307	0.9739	0.0503

selection algorithms, and regression models used, 64 % of the models constructed with feature selection demonstrate higher success and better fit compared to those built with the original feature set. Furthermore, even with a reduced number of features, the remaining models do not show a significant increase in prediction error.

Table 5 provides a comparison between the results obtained in this study and those from previous research utilizing the same dataset. While Serbest et al. [21] achieved notably high predictive accuracy across all joints, their use of 22 input features may be perceived as a drawback. In contrast, our study, despite employing fewer input features, demonstrates comparable levels of predictive success to the study by Mansour et al. [18]. According to Tables 5 and it is seen that the proposed EO-ETR

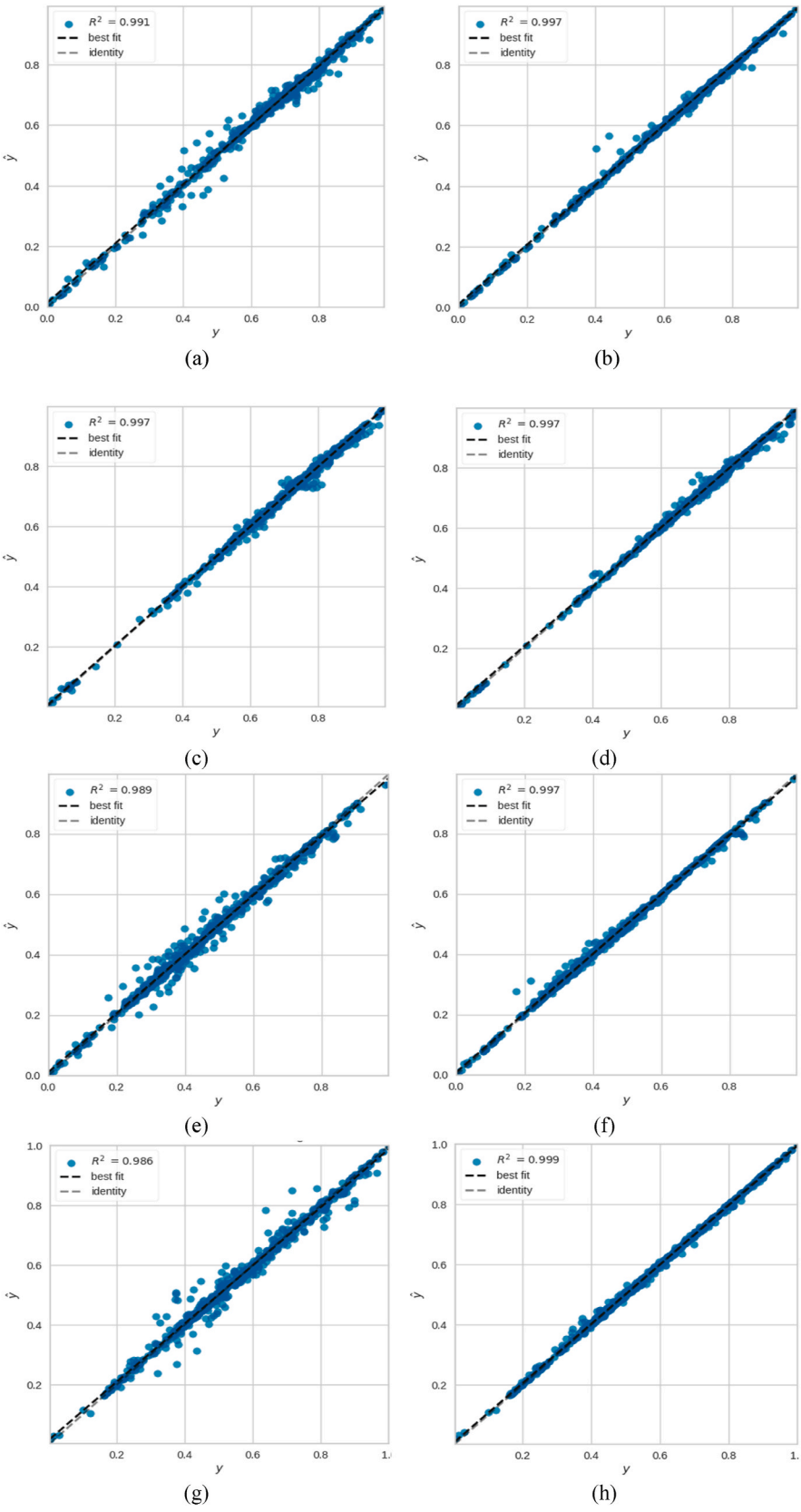

Fig. 4. Prediction Errors of (a) ETR for original features of ankle dataset (b) ETR for EO selected features of ankle dataset (c) ETR for original features of knee dataset (d) ETR for EO selected features of knee dataset (e) ETR for original features of hip dataset (f) ETR for MPA selected features of hip dataset (g) ETR for original features of neck dataset (h) ETR for EO selected features of neck dataset.

Table 5
R^2 value comparison of the results with other STS studies for same dataset.

Ref.	Algorithm	Features	R^2			
			Ankle	Knee	Hip	Neck
Our study[a]	**EO-ETR**	**Total selected:4/25**	**0.996**	**0.996**	**0.997**	**0.997**
	EO-ETR	**Total selected: 3/25**				
	MPA-ETR	**Total selected: 3/25**				
[16]	SVM	Total selected: 19/22, Features: time and three joints information	0.908	0.940	0.930	N/A
	LR		0.869	0.951	0.900	N/A
	RF		0.989	0.989	0.989	N/A
	DT		0.973	0.983	0.955	N/A
	DNN		0.957	0.970	0.961	N/A
	LSTM		0.998	0.999	0.998	N/A
	CNN		0.991	0.999	0.995	N/A
[16]	SVM	Total selected: 7/22, Features: time and one joints information	0.743	0.902	0.833	N/A
	LR		0.126	0.834	0.631	N/A
	RF		0.991	0.995	0.974	N/A
	DT		0.975	0.986	0.960	N/A
	DNN		0.763	0.912	0.823	N/A
	LSTM		0.996	0.997	0.996	N/A
	CNN		0.968	0.992	0.991	N/A
[21]	ANN	Total selected: 22/22, Features: Subject number, time, definition number of (ankle, knee, hip, neck), degree (ankle, knee, hip, neck), height (ankle, knee, hip, neck), mass (ankle, knee, hip, neck), moment of inertia (ankle, knee, hip, neck)	1	1	0.999	1

[a] Results with the highest R^2 value.

Table 6
R^2 value comparison of the results with other studies (different movement). SVM; support vector machine, CNN; Convolutional Neural Networks.

Ref.	Movement	Algorithm	Features	R^2			
				Ankle	Knee	Hip	Neck
[25]	Walking	WNN	18	0.880[b]			N/A
[24]	Running	ANN	14	0.955	0.954	0.927	N/A
[49]	Squat	SVM-LSTM	99	0.864	0.828	0.921	N/A
[23]	Walking	FNN	30	0.902[b]			N/A
Our study[a]	STS	EO-ETR	4	0.996	0.996	0.997	0.997
		EO-ETR	3				
		MPA-ETR	3				

[a] Results with the highest R^2 value.
[b] Mean value for all joints.

and MPA-ETR algorithms achieve higher success with fewer features for four different joints (ankle, knee, hip and neck) in the same data set. For the ankle joint, the EO-ETR algorithm successfully predicted 0.996 R^2 by selecting 4 features out of 25 features. The R^2 success of other algorithms in the literature is 0.998 with 19 features for LSTM, 0.996 with 7 features, and 1.0 with 22 features for ANN. For the knee joint, the EO-ETR algorithm achieved an R^2 of 0.996 by selecting 3 out of 25 features. In comparison, other algorithms in the literature have reported R^2 values of 0.999 with 19 features for LSTM and CNN, 0.997 with 7 features for LSTM, and 1.0 with 22 features for ANN. For the hip joint, the MPA-ETR algorithm achieved an R^2 of 0.997 by selecting 3 out of 25 features. In comparison, other algorithms in the literature have reported R^2 values of 0.998 and 0.996 with 19 features and 7 features for LSTM, respectively, and 0.999 with 22 features for ANN. In the neck joint, the EO-ETR algorithm selected 4 features out of 25 and predicted them with a success of 0.997 R^2, while the ANN algorithm showed a success of 1.0 R^2 with 22 features.

Table 6 provides an R^2 performance evaluation comparing the outcomes of various studies on joint moment estimation across different movements and exercises. Analysis of these results reveals a significant reduction in the number of features used in our study, alongside notably enhanced accuracy in joint moment estimation. Notably, our study achieved high accuracy in estimating the neck joint moment, which features a distinct movement pattern compared to lower extremity joints. These findings underscore the efficacy of the algorithms proposed in this study for accurately estimating joint moments during the STS movement, even with a limited number of features.

Recent studies in the literature, the field of joint moment estimation during STS movements has seen significant advancements, driven by the integration of ML and meta-heuristic optimization algorithms. Traditional methods, while accurate, often suffer from limitations in practical application due to their complexity and the need for extensive sensor data. Conversely, the application of ML offers a promising alternative, providing enhanced accuracy and efficiency. However, this shift has sparked debates regarding the generalizability and robustness of these new methods across diverse populations and movement conditions. Recent achievements, such as the development of the Equilibrium Optimizer [42] and the Marine Predators Algorithm [37], have significantly improved feature selection for joint moment prediction. Additionally, advances in integrating ML with biomechanical modeling [19] and the establishment of novel datasets and benchmarking standards [31] have propelled the field forward.

This study aims to leverage these recent innovations to enhance the accuracy and clinical applicability of joint moment predictions during STS movements, addressing the current controversies and highlighting the novelty of these advancements. Our conclusion is that meta-heuristic algorithms select a smaller number of features that best represent the dataset. Overall, although the proposed algorithms did not achieve the highest accuracy in some of the datasets, they significantly reduced the number of features.

4. Conclusions

The feature selection methods MRFO, MPA and EO algorithms are

powerful methods that effectively identify meaningful features while eliminating irrelevant ones. In this study, we aimed to improve the prediction performance of ML through optimized feature selection with metaheuristic algorithms. We investigate different meta-heuristic optimization algorithms (MRFO, MPA, EO) and different ML methods (DTR, RFR, ETR, XGBoost Regression) to predict the moments occurring in the neck, hip, knee, and ankle joints during STS.

According to the experiment's findings, with a small number of features (2–4) joint moments can be estimated with high accuracy. The comparison of the proposed algorithms with other methods and previous work in feature selection demonstrates the MPA algorithm's high capability in selecting the best features, thereby enhancing joint moment prediction. While the performance of the EO-ETR offers good performance for ankle, knee and neck with 0.996, 0.996 and 0.997 R2 respectively, MPA-ETR method is achieved good performance with 0.997 R2 for hip. It is anticipated that the results obtained from this study will benefit research teams consisting of experts from different disciplines working on clinical studies and biomechanical analyses of STS motion. The most important of these benefits are to perform motion analyses with a simpler set-up and to calculate fewer mechanical parameters.

It is evident that many AI methods developed for predicting kinematic and kinetic changes in human movements utilize a large number of inputs/features, complicating the data collection process during clinical trials. This study, aimed at reducing the number of features using meta-heuristic optimization algorithms, represents a pioneering effort in the literature. The methods introduced in this study can also be applied to predict kinematic and kinetic changes in human movements beyond STS.

In the future work, we are planning to develop hybrid metaheuristic algorithm with the other machine learning algorithms to predict joint moments during STS.

CRediT authorship contribution statement

Ekin Ekinci: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Conceptualization. **Zeynep Garip:** Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Conceptualization. **Kasim Serbest:** Writing – review & editing, Writing – original draft, Validation, Supervision, Methodology, Conceptualization.

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] W.G. Janssen, H.B. Bussmann, H.J. Stam, Determinants of the sit-to-stand movement: a review, Phys. Ther. 82 (9) (2002) 866–879.

[2] S. Yoshioka, A. Nagano, R. Himeno, S. Fukashiro, Computation of the kinematics and the minimum peak joint moments of sit-to-stand movements, Biomed. Eng. Online 6 (2007) 1–14.

[3] H.R. Yamasaki, H. Kambara, Y. Koike, Dynamic optimization of the sit-to-stand movement, J. Appl. Biomech. 27 (4) (2011) 306–313.

[4] C.S.N. da Costa, G. Savelsbergh, N.A.C.F. Rocha, Sit-to-stand movement in children: a review, J. Mot. Behav. 42 (2) (2010) 127–134.

[5] E. Warmerdam, J.M. Hausdorff, A. Atrsaei, Y. Zhou, A. Mirelman, K. Aminian, W. Maetzler, Long-term unsupervised mobility assessment in movement disorders, Lancet Neurol. 19 (5) (2020) 462–470.

[6] K. Serbest, Effects of weight gaining to lower limb joint moments: a gender-specific sit-to-stand analysis, Biomedical Engineering/Biomedizinische Technik 67 (6) (2022) 481–489.

[7] K. Han, K. Bae, N. Levine, J. Yang, J.S. Lee, Biomechanical effect of foot orthoses on rearfoot motions and joint moment parameters in patients with flexible flatfoot, Med. Sci. Mon. Int. Med. J. Exp. Clin. Res.: international medical journal of experimental and clinical research 25 (2019) 5920.

[8] K. Liu, J. Yan, Y. Liu, M. Ye, Noninvasive estimation of joint moments with inertial sensor system for analysis of STS rehabilitation training, Journal of healthcare engineering 2018 (2018).

[9] W. Blajer, K. Dziewiecki, Z. Mazur, Multibody modeling of human body for the inverse dynamics analysis of sagittal plane movements, Multibody Syst. Dyn. 18 (2007) 217–232.

[10] A. Rajagopal, C.L. Dembia, M.S. DeMers, D.D. Delp, J.L. Hicks, S.L. Delp, Full-body musculoskeletal model for muscle-driven simulation of human gait, IEEE Trans. Biomed. Eng. 63 (10) (2016) 2068–2079.

[11] E.M. Arnold, S.R. Ward, R.L. Lieber, S.L. Delp, A model of the lower limb for analysis of human movement, Ann. Biomed. Eng. 38 (2010) 269–279.

[12] S.L. Delp, F.C. Anderson, A.S. Arnold, P. Loan, A. Habib, C.T. John, D.G. Thelen, OpenSim: open-source software to create and analyze dynamic simulations of movement, IEEE Trans. Biomed. Eng. 54 (11) (2007) 1940–1950.

[13] U. Trinler, H. Schwameder, R. Baker, N. Alexander, Muscle force estimation in clinical gait analysis using AnyBody and OpenSim, J. Biomech. 86 (2019) 55–63.

[14] T.R. Noviandy, A. Maulana, G.M. Idroes, T.B. Emran, T.E. Tallei, Z. Helwani, R. Idroes, Ensemble machine learning approach for quantitative structure activity relationship based drug discovery: a Review, Infolitika Journal of Data Science 1 (1) (2023) 32–41.

[15] N. Absar, B. Mamur, A. Mahmud, T.B. Emran, M.U. Khandaker, M.R.I. Faruque, B. A. Elkhader, Development of a computer-aided tool for detection of COVID-19 pneumonia from CXR images using machine learning algorithm, Journal of Radiation Research and Applied Sciences 15 (1) (2022) 32–43.

[16] Q. Yang, B. Li, J. Tang, X. Cui, Y. Wang, X. Li, F. Zhu, Consistent gene signature of schizophrenia identified by a novel feature selection strategy from comprehensive sets of transcriptomic data, Briefings Bioinf. 21 (3) (2020) 1058–1068.

[17] Q. Yang, B. Li, S. Chen, J. Tang, Y. Li, Y. Li, F. Zhu, MMEASE: online meta-analysis of metabolomic data by enhanced metabolite annotation, marker selection and enrichment analysis, J. Proteonomics 232 (2021) 104023.

[18] M. Mansour, K. Serbest, M. Kutlu, M. Cilli, Estimation of lower limb joint moments based on the inverse dynamics approach: a comparison of machine learning algorithms for rapid estimation, Med. Biol. Eng. Comput. 61 (12) (2023) 3253–3276.

[19] L. Zhang, Z. Li, Y. Hu, C. Smith, E.M.G. Farewik, R. Wang, Ankle joint torque estimation using an EMG-driven neuromusculoskeletal model and an artificial neural network model, IEEE Trans. Autom. Sci. Eng. 18 (2) (2020) 564–573.

[20] C. Fang, B. He, Y. Wang, J. Cao, S. Gao, EMG-centered multisensory based technologies for pattern recognition in rehabilitation: state of the art and challenges, Biosensors 10 (8) (2020) 85.

[21] K. Serbest, M.T. Ozkan, M. Cilli, Estimation of joint torques using an artificial neural network model based on kinematic and anthropometric data, Neural Comput. Appl. 35 (17) (2023) 12513–12529.

[22] M.S.B. Hossain, Z. Guo, H. Choi, Estimation of lower extremity joint moments and 3D ground reaction forces using IMU sensors in multiple walking conditions: a deep learning approach, IEEE Journal of Biomedical and Health Informatics 27 (6) (2023) 2829–2840.

[23] M. Mundt, A. Koeppe, S. David, T. Witter, F. Bamer, W. Potthast, B. Markert, Estimation of gait mechanics based on simulated and measured IMU data using an artificial neural network, Front. Bioeng. Biotechnol. 8 (2020) 41.

[24] B. Xiong, N. Zeng, H. Li, Y. Yang, Y. Li, M. Huang, Y. Zhang, Intelligent prediction of human lower extremity joint moment: an artificial neural network approach, IEEE Access 7 (2019) 29973–29980.

[25] M.M. Ardestani, X. Zhang, L. Wang, Q. Lian, Y. Liu, J. He, Z. Jin, Human lower extremity joint moment prediction: a wavelet neural network approach, Expert Syst. Appl. 41 (9) (2014) 4422–4433.

[26] Z. Garip, E. Ekinci, M.E. Çimen, A Comparative Study of Optimization Algorithms for Feature Selection on ML-Based Classification of Agricultural Data, Cluster Computing, 2023, pp. 1–22.

[27] Ş. Ay, E. Ekinci, Z. Garip, A comparative analysis of meta-heuristic optimization algorithms for feature selection on ML-based classification of heart-related diseases, J. Supercomput. (2023) 1–30.

[28] F. Atban, E. Ekinci, Z. Garip, Traditional machine learning algorithms for breast cancer image classification with optimized deep features, Biomed. Signal Process Control 81 (2023) 104534.

[29] W. Zhao, Z. Zhang, L. Wang, Manta ray foraging optimization: an effective bio-inspired optimizer for engineering applications, Eng. Appl. Artif. Intell. 87 (2020) 103300.

[30] U. Agrawal, V. Rohatgi, R. Katarya, Normalized Mutual Information-based equilibrium optimizer with chaotic maps for wrapper-filter feature selection, Expert Syst. Appl. 207 (2022) 118107.

[31] M. Cilli, K. Serbest, E. Kayaoglu, The effect of body weight on joint torques in teenagers: Investigation of sit-to-stand movement, Clin. BioMech. 83 (2021) 105288.

[32] D. Brown, Tracker video analysis and modeling tool, Retrieved from http://www.opensourcephysics.org/items/detail.cfm?ID=7365, , 2008.

[33] M.R. Yeadon, The simulation of aerial movement—II. A mathematical inertia model of the human body, J. Biomech. 23 (1) (1990) 67–74.

[34] S. Li, X. Kong, L. Yue, C. Liu, M.A. Khan, Z. Yang, H. Zhang, Short-term electrical load forecasting using hybrid model of manta ray foraging optimization and support vector regression, J. Clean. Prod. 388 (2023) 135856.

[35] E.H. Nezhad, R. Ebrahimi, M. Ghanbari, Fuzzy Multi-objective allocation of photovoltaic energy resources in unbalanced network using improved manta ray foraging optimization algorithm, Expert Syst. Appl. 234 (2023) 121048.

[36] Z. Tao, C. Zhang, J. Xiong, H. Hu, J. Ji, T. Peng, M.S. Nazir, Evolutionary gate recurrent unit coupling convolutional neural network and improved manta ray

foraging optimization algorithm for performance degradation prediction of PEMFC, Appl. Energy 336 (2023) 120821.
[37] A. Faramarzi, M. Heidarinejad, S. Mirjalili, A.H. Gandomi, Marine predators algorithm: a nature-inspired metaheuristic, Expert Syst. Appl. 152 (2020) 113377.
[38] X. Chen, X. Qi, Z. Wang, C. Cui, B. Wu, Y. Yang, Fault diagnosis of rolling bearing using marine predators algorithm-based support vector machine and topology learning and out-of-sample embedding, Measurement 176 (2021) 109116.
[39] D. Yousri, H.M. Hasanien, A. Fathy, Parameters identification of solid oxide fuel cell for static and dynamic simulation using comprehensive learning dynamic multi-swarm marine predators algorithm, Energy Convers. Manag. 228 (2021) 113692.
[40] M. Abdel-Basset, D. El-Shahat, R.K. Chakrabortty, M. Ryan, Parameter estimation of photovoltaic models using an improved marine predators algorithm, Energy Convers. Manag. 227 (2021) 113491.
[41] A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: a novel optimization algorithm, Knowl. Base Syst. 191 (2020) 105190.
[42] M. Zhang, J.S. Wang, Y. Liu, H.M. Song, J.N. Hou, Y.C. Wang, M. Wang, Multi-objective optimization algorithm based on clustering guided binary equilibrium optimizer and NSGA-III to solve high-dimensional feature selection problem, Inf. Sci. 648 (2023) 119638.
[43] X. Sun, M.J.C. Opulencia, T.P. Alexandrovich, A. Khan, M. Algarni, A. Abdelrahman, Modeling and optimization of vegetable oil biodiesel production with heterogeneous nano catalytic process: multi-layer perceptron, decision regression tree, and K-Nearest Neighbor methods, Environ. Technol. Innovat. 27 (2022) 102794.
[44] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32.
[45] B. Babar, L.T. Luppino, T. Boström, S.N. Anfinsen, Random forest regression for improved mapping of solar irradiance at high latitudes, Sol. Energy 198 (2020) 81–92.
[46] P. Geurts, D. Ernst, L. Wehenkel, Extremely randomized trees, Mach. Learn. 63 (2006) 3–42.
[47] Y. Ren, Z. Lv, Z. Xu, Q. Wang, Z. Wang, Slurry-ability mathematical modeling of microwave-modified lignite: a comparative analysis of multivariate non-linear regression model and XGBoost algorithm model, Energy (2023) 128143.
[48] E. Ekinci, B. Özbay, S.İ. Omurca, F.E. Sayın, İ. Özbay, Application of machine learning algorithms and feature selection methods for better prediction of sludge production in a real advanced biological wastewater treatment plant, J. Environ. Manag. 348 (2023) 119448.
[49] S. Chae, A. Choi, H. Jung, T.H. Kim, K. Kim, J.H. Mun, Machine learning model to estimate net joint moments during lifting task using wearable sensors: a preliminary study for design of exoskeleton control system, Appl. Sci. 11 (24) (2021) 11735.