A Deep Learning Based Approach for Foot Placement Prediction with Attention Mechanism

Zhengqi Dong
Boston University College of Engineering
110 Cummington Mall, Boston, MA
dong760@bu.edu

Abstract

Predicting the next foot placement of human walking or running activity is an important task for developing a comfortable and intelligent walking aid robot. The previous study mainly focuses on designing the walking aid robot with better mechanical support or adding some traditional machining learning approaches that require extensive manual feature selection to generate the foot placement probability grid map. Those approaches can be difficult to generalize to various complicated scenarios in human daily life activities. Therefore, an attention mechanism with several state-of-the-art neural network techniques was investigated for improving the generalization ability and accuracy of the foot placement prediction task. In this study, we implemented and compared four different neural network architectures for foot placement prediction, and the experimental result are obtained and showed a clear improvement in the effectiveness of adding an attention mechanism to the novel network architecture that only consists of Long Short-Term Memory (LSTM) layer or Bidirectional LSTM (Bi-LSTM) layer. An experimental comparison found that the MAE, MSE, and RMSE prediction error has reduced by about 8.64%, 16.97%, and 8.88% after adding the attention layer meanwhile this performance does not correlate with the model size. Besides, our result also shows that simply replacing the LSTM with the Bi-LSTM layer would not improve the performance and can be harmful instead.

1. Introduction

Walking and running are important gaits of terrestrial locomotion that human exercises every day, and this activity is majorly supported by our lower-limb body. However, many elderly people and patients who were neurologically injured by stroke, spinal cord injury, weakness of skeletal muscles, and Parkinson's disease have difficulties in coordinating their body and walking normally [1], [2]. Furthermore, they were subjected to a high risk of falling in many common daily living activities during locomotion, including stepping over obstacles, turning, and stair climbing [3], [4]. Recently, many intelligent walking-aid robots (e.g., prostheses, orthoses, exoskeletons devices, extra robots) had been designed for helping elderly and dysfunctional walking patients with physical movement assistance and rehabilitation. For example, the Walking-aid Cane robot [5] uses a center of pressure (COP) method and Dubois' fuzzy possibility on the cane robot to predict and prevent the potential falling event, and a Bayesian inference-based foot placement prediction model is proposed to model the foot-placement probability map [6].

In this project, for pushing the limit for building a more intelligent exoskeleton device, we implemented and compared four different neural network architectures for verifying the validity of the deep learning approach and measuring the effectiveness of adding an attention mechanism to the LSTM network architecture on food placement prediction tasks. Moreover, a simple data collection plan is created and conducted for preparing the training data for this study.

The rest of the paper is organized as follows. In Section 2, we introduced several gait phases that a completed gait cycle was made up of. The data collection and gait segmentation are introduced in Section 3. Section 4 introduces several most popular neural network models and the four network architectures that we implemented in this study. Sections 5 and 6 show the result and discuss certain limitations and drawbacks to be improved in the future. Finally, Section 7 concludes the paper.

2. Literature Review

2.1. Gait Phase Analysis

Human walking is a cyclical and repetitive motion. The entire walking cycle can be divided into stance and swing phases, shown in Figure 1 [8]. Gait phase percentage differs depending on individual walking speed and gait personal gait characteristics. In general, the stance phase and swing phase takes about 60% and 40% of the gait cycle, respectively. A conventional stance phase can be divided into initial contact (heel strike), loading response, mid-stance, terminal stance, and pre-swing period. Two double suppose period occurs at the beginning and the end of the stance phase, where two feet are in contact with the supporting surface. The swing phase can be subdivided

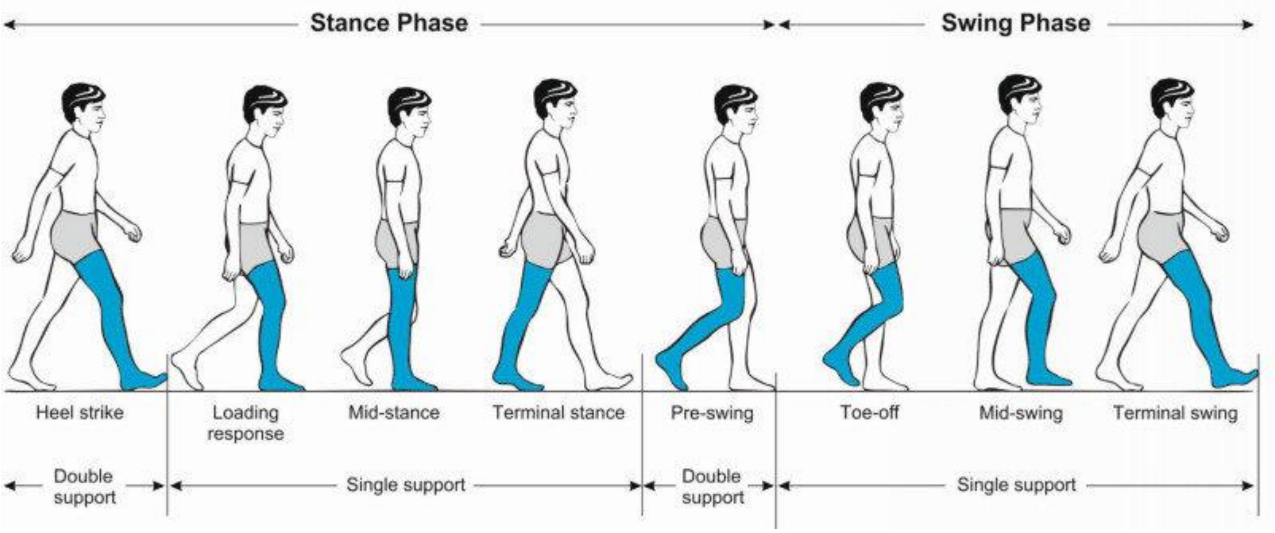


Figure 1: Schematic diagram of a complete gait cycle on a healthy human [8].

into the initial swing (toe-off), mid-swing, and terminal swing. In this project, the first 50% of the swing phase (starting from toe-off to mid-swing event) will be used as training data, and the cartesian position of the heel strike event at the next gait cycle, where the right foot initial contacts with the support surface, will be used as the ground truth label.

2.2. Gait Features

Human gait features can be categorized into three components [9]: spatiotemporal, kinematics, and kinetics features. 1) Spatiotemporal features are the most commonly used feature to describe gait patterns, including gait cycle, stance phase, swing phase, double limb support, single limb support, step duration, stride length, step length, step width, and foot progression angle. 2) Kinetic features involved the study of dynamic force on walking movements, such as measuring the ground reaction force(GRF) on the hip, knee, and ankle joints. 3) Kinematic feature involves the study of walking motion without taking the force into account, such as analyzing the position, displacement, velocity, and acceleration of certain body areas or join (e.g., ankle, knee, hip, pelvis, or heel) [9], [10]. As we can see in [6], [9], most of the research that had been conducted on the traditional machine learning method requires a lot of manual feature selection and extraction steps before the training process. A deep learning approach not only can remove those cumbersome processes but is also capable of learning all useful features more effectively during the training phase.

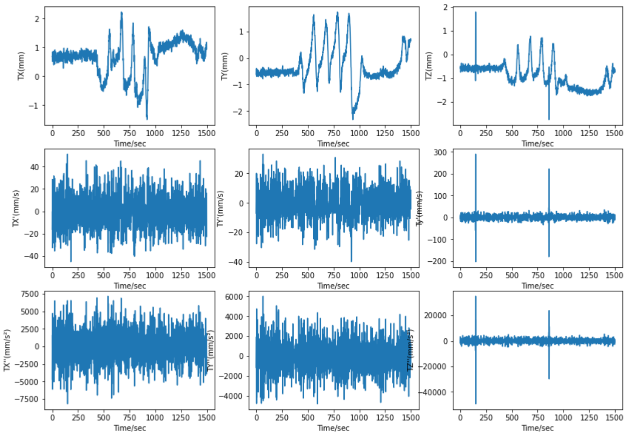


Figure 2: EDA plot of normalized displacement, velocity, and acceleration along the x-y-z axis respectively on the first 3000 timeframes (~300 sec) of HAR Dataset.

3. Dataset

3.1. Data Preparation

The Bath Natural Environment HAR dataset in [7] was collected with five Suunto Movesense wearable IMU sensors attached to ankles(two), hips(two), and chests of 22 healthy and injury-free adults with various ages (mean 29, std 10) and gender (17M, 5F) while walking across six different locomotive activity: Walking (179min, 9438 steps), Stair Ascent (23 min, 1286 steps), Stair Descent (20min, 1280 steps), Ramp Ascent (12min, 656 steps), Ramp Descent (13 min, 754 steps), and Stop (20 min). After some data processing, we plot the displacement, velocity, and acceleration along the x-y-z axis in Figures 2 and 3.

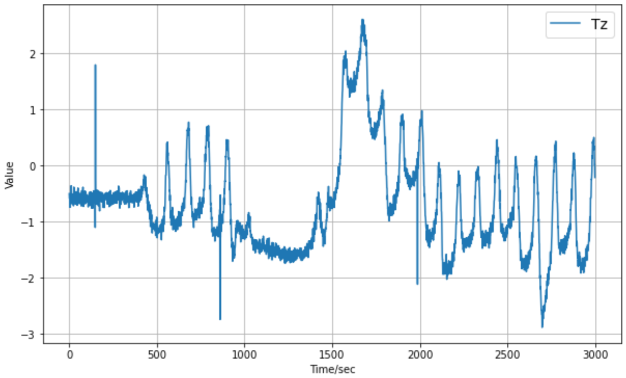


Figure 3: Zoom-in plot of displacement along the z-axis.

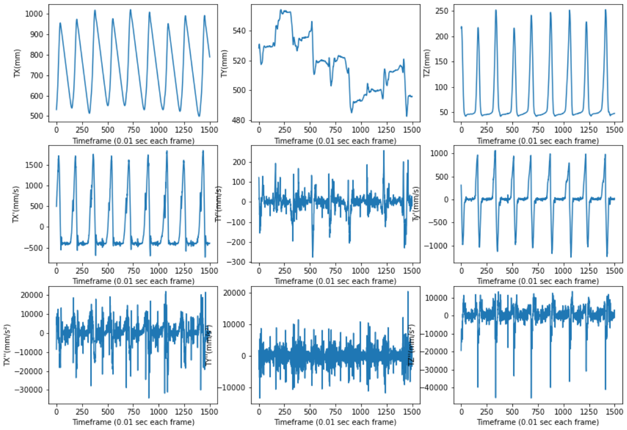


Figure 4: EDA plot of displacement, velocity, and acceleration along the x-y-z axis respectively on the first 1500 timeframe(~150 sec) of Motion-Capture Dataset

Owning to a lot of noise in the IMU data, it is infeasible to perform the gait segmentation, segmenting the whole sequential data into each gait cycle that starts from toe-off to heel-strike event, on all walking activities across different participants. On the other hand, since we need the gait trajectory and ground truth position at each heel-strike event for training the model, we decided to use our motion capture device, Vicon Nexus 2.12 system, to collect the data for better training results, which will be named as Vicon dataset in this project. The data was collected with a marker placed on the right shank of a healthy normal human and a walking speed of 0.4m/s for 3 min at 100 HZ, and the Exploratory Data Analysis (EDA) plots on a subsequence of 1500 time steps are shown in Figure 4. The details about the data collection plan and environment setup can be seen in Appendix A.

3.2. Gait Segmentation

Gait segmentation is a crucial step in gait analysis and foot-placement prediction. In general speaking, gait segmentation involves the detection of the following three sub-phases of stance: heel contact, flat foot contact, push-off (or heel off), and limb swing [11]. However, in this project, for simplifying the process, we follow the algorithm described in [6], where a gait cycle contains three events (toe-off, mid-swing, and heel-strike) in the swing phase and must happen in the sequence shown in Figure 1. All gait cycles that do not meet this requirement will be discarded. The gait segmentation result of displacement along the z-axis is shown in Figure 5. The red dashed line indicates the toe-off event, where the foot completely leaves the ground, and the following three cycles indicate three sub-phases (toe-off, mid-swing, and heel-strike) required for a completed gait cycle. The position of the heel-strike event will be used as the ground truth position for training later.

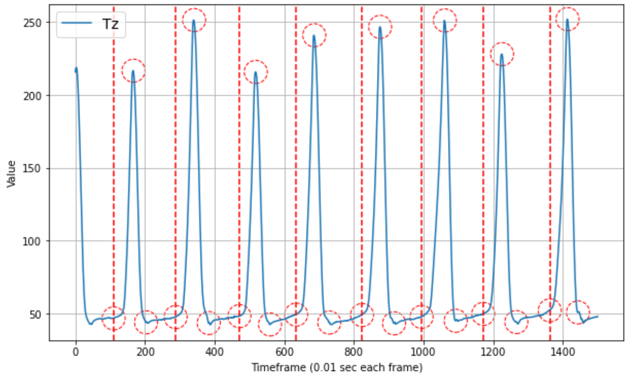


Figure 5: Gait segmentation result of displacement on the z-axis

A similar algorithm for gait segmentation can be referred to [11] and a deep learning approach [12], [13] that might be more reliable on various human gait and generalize better in some challenging conditions involving multiple activities (e.g., running, walking upstairs and downstairs, walking uphill and downhill).

4. Methodology

4.1. Neural Network

The conventional ML approach requires manual gait feature selection. As an example of the Bayesian inference approach shown in [6], this process can be time-consuming and requires a practitioner has a solid understanding of gait features and underlying algorithms. Thus, many researchers had turned their focus to the success of deep learning, such as [14], Zhang. K used CNN based unsupervised cross-subject adaptation network for predicting the human locomotion intent and activity states of the subject with over 90% accuracy. The crux of using a neural network for foot placement prediction was stimulated by the well-known universal approximation theorem [15], a neural network with a nonpolynomial

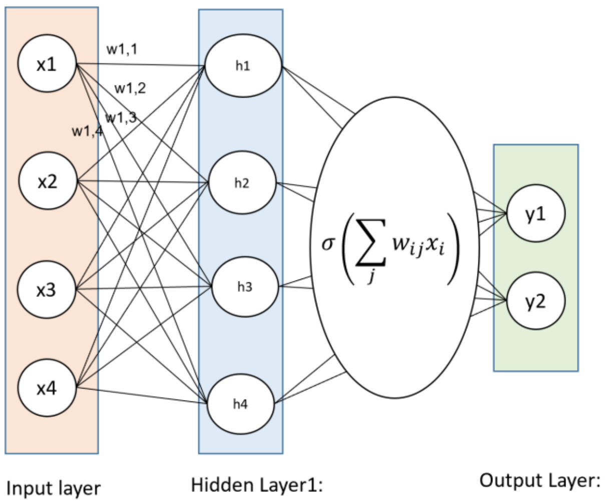


Figure 6: An example MLP neural-network structure with a single hidden layer (bias is omitted for brevity).

activation function can approximate any continuous function.

A standard fully connected multiplayer network consists of the following characteristics: 1) a vector of “weights” denoted $W = w_1, w_2, \dots, w_n$, 2) an activation function $\sigma: R \to R$, $\hat{y} = \sigma(\boldsymbol{w}^\top \boldsymbol{x} + b) = \sigma(\sum_i w_i x_i + b) = \sigma(x_1 w_1 + x_2 w_2 + \cdots + x_n w_n + b)$, such as Sigmoid, ReLU, Tanh; 3) an error function, $E = \mathcal{L}(Y, \hat{Y})$ that measure the difference between predicted results and desired results, such as Mean Absolute Error, Mean Squared Error, Categorical Cross entropy Loss, [15]–[17];

The conventional training process takes a training dataset $D = \{(\boldsymbol{X}, \boldsymbol{Y}) = (x_1, y_1), \dots, (x_N, y_N)\}$ and feed into the network to learn a predictive model $\hat{y} = f_\theta(x)$ that is parameterized by θ, by solving

$$W^* = \arg\min_W \mathcal{L}(\mathcal{D}; W, \omega)$$

, where $\mathcal{L}$ is the loss function, and ω is a predefine assumption about the learning processes, such as the optimizer, learning rate, and activation function that is chosen for the network [16]–[19].

For example, a three-layer of multilayer perceptron’s (MLP) network, shown in Figure 2, contains four neurons as the input layer, four neurons in the hidden layer, and two neurons in the output layer. Assuming sigmoid is used as the activation function,

$$\widehat{y_1} = \sigma(\boldsymbol{x}, \boldsymbol{w}) = \frac{1}{1 + e^{\boldsymbol{w}^T \boldsymbol{x}}}$$

and mean square error are used as the loss function, that is

$$E(\boldsymbol{W}) = \frac{1}{2}\sum_k (\widehat{y_k} - y_k)^2$$

Then, with a typical mini-batch gradient descent algorithm as the objective function optimizer, we will have the following algorithm for training the model:

```
Initialize weights randomly $\boldsymbol{W_i} \sim N(0, \sigma^2)$
Loop until convergence:
    for $i \leftarrow 1$ $to$ $\left[\frac{data\_size}{\text{batch\_size}}\right]$ do:
        for $j \leftarrow 1$ $to$ $batch\_size$ do:
            Compute Gradient, $\nabla J(\boldsymbol{W_i}) = \nabla J_j(\boldsymbol{W_i}) + \nabla J(\boldsymbol{W_i})$
        $\nabla J(\boldsymbol{W_i}) = \frac{1}{batch\_size}\nabla J(\boldsymbol{W_i})$
        Update parameter, $\boldsymbol{W_{i+1}} \leftarrow \boldsymbol{W_i} - \eta\nabla J(\boldsymbol{W_i})$
Return weights
```

With much evidence shown in [9], [14], [20], [21], the neural network-based approach, especially deep neural network with convolutional neural network (CNN) module, outperformed other traditional machine learning methods in the field of gait analysis, including SVM, Decision tree, LDA, Bayesian classifier, Random forest, kNN. Thus, we believe that, with a delicate design of deep learning architecture, the accuracy of foot placement prediction published in [6] can also be improved significantly.

4.2. Recurrent Neural Network (RNN)

Using a traditional neural network model (e.g, Dense Neural Network (DNN), or Convolutional Neural Network (CNN)) to handle gait trajectory with variable dimensions input data can be a difficult task. Recurrent neural network (RNN) is a special kind of deep learning model, and it has been widely used in many areas related to sequence data processing, including machine translation[22], speech recognition[23], image captioning [24], and video analysis task[25]. A typical RNN network is shown in Figure 7, but there are many ways to design the recurrent connections, including a recurrent network without output, sequence-input, and single-output, sequence-input, and sequence-output, RNN with teacher forcing, Bidirectional RNN, etc [17]. The looping structure designed within RNNs makes it well-suited for processing and modeling sequential data with a variable number of input and output dimensions. However, owing to the vanishing and exploding gradient problem mentioned in [26], it is challenging to train RNNs networks to solve problems that require learning long-term dependencies. As many research applications [27]–[29] had shown that an LSTM network can be a more effective approach in long-range reasoning for nonlinear sequential data with various lengths.

4.3. Long Short Ter Memory (LSTM)

Long Short-Term Memory (LSTM) network was first introduced by Hochreiter & Schmidhuber in 1997 [30], in which the problem of training conventional RNNs network was solved by introducing a more effective

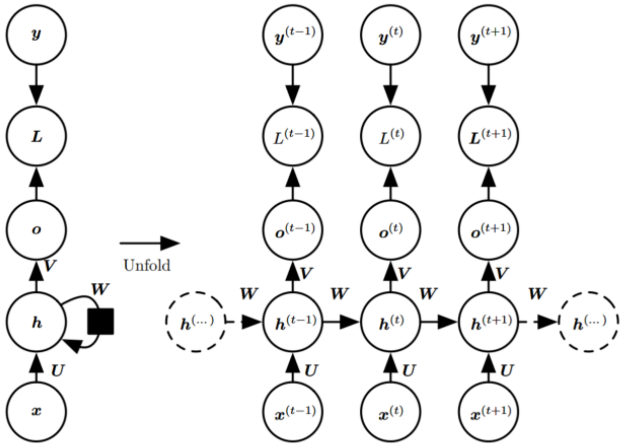


Figure 7: A typical RNN network graph that maps a sequence of input x to a corresponding sequence of output o, and then the difference between output o and target output y will be measured by a loss function $\mathcal{L}(y, \mathrm{o})$ [17].

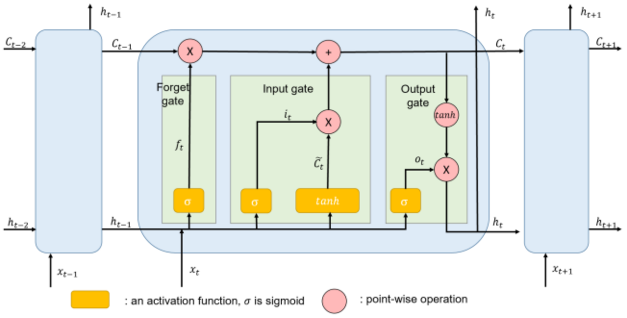


Figure 8: A typical LSTM network structure [34]

gradient learning method with some gating mechanisms that can store, retrieve, and remove information over a longer sequence. In around 2014, many researchers contributed to refining the LSTM network and achieved greater success in language processing-related tasks [31]–[33], and now it was widely known.

A typical LSTM network structure is shown in Figure 8. Two vectors (C_t, h_t) need to be maintained at each point of the network, where C_t is the cell state vector and h_t is the output vector. Three gate blocks are used to control the flow of data: forget gate, input gate, and output gate, and their mathematical expressions are shown in equations (1) to (3) respectively.

The LSTM network use forget gate to decide what information will be removed. A sigmoid function is used here to output a value between zero and one to keep the network differentiable, where zero means removing everything from the last output h_{t-1}, and one means letting everything pass through [33], [34].

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

The input gate decides what new information we want to add to the next cell state C_t, and the candidate's new state $\widetilde{C}_t$ will be multiplied with a scaling factor i and added to the last cell state C_{t-1} (the * here indicates element-wide multiplication) [33], [34].

$$\begin{aligned} i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\ \widetilde{C}_t &= \tanh(W_C[h_{t-1}, x_t] + b_C) \\ C_t &= f_t * C_{t-1} + i_t * \widetilde{C}_t \end{aligned} \tag{2}$$

The output gate decides what information is used as output. A $tanh$ activation function is used on the updated state value C_t to ensure the value is ranged between -1 and 1 [33], [34].

$$\begin{aligned} \mathrm{o_t} &= \sigma(\mathrm{W_o}[\mathrm{h_{t-1}}, \mathrm{x_t}] + \mathrm{b_o}) \\ \mathrm{h_t} &= \mathrm{o_t} * \tanh(\mathrm{C_t}) \end{aligned} \tag{3}$$

4.4. Bidirectional Long Short Ter Memory (Bi-LSTM)

LSTM is a unidirectional network and only can preserves information in the past, whereas Bidirectional LSTM (Bi-STM) operates on the input sequence from two directions (start-end and end-start). This approach not only allows the network to take information from both the future and past into consideration in predicting the result but also has the advantage of learning data sequences with long-time dependencies [35]. The study result in [35] has shown that Bi-LSTM performs significantly better than any other neural network on the task of framewise phoneme classification, including RNN, Bidirectional RNN (BRNN), and LSTM. Besides, some latest gait analysis-related research has already adopted this method in developing their network architecture and shown a good performance increase in accuracy [28], [36], [37].

4.5. Attention Mechanism

The idea of the attention mechanism was first introduced by Bahdanau [38] to address the bottleneck issue of the fixed-length vector that is being used in the encoder-decoder architecture for neural machine translation (NMT). As this approach achieved great success in improving the performance of the NMT model, the attention mechanisms started to gain more attention and many variants have been created [39], [40]. The underlying details in [38] can be quite complicated to demonstrate here, but the idea is simple. In traditional LSTM-based NMT, the encoder-decoder architecture is often used, where the encoder will encode the whole input sequence into a fixed-length vector, and a translation with the target language will be decoded as output. With the attention mechanism, the encoder will pass all the hidden states to the decoder, and the decoder will enhance the weight of the selected subset of words that is more

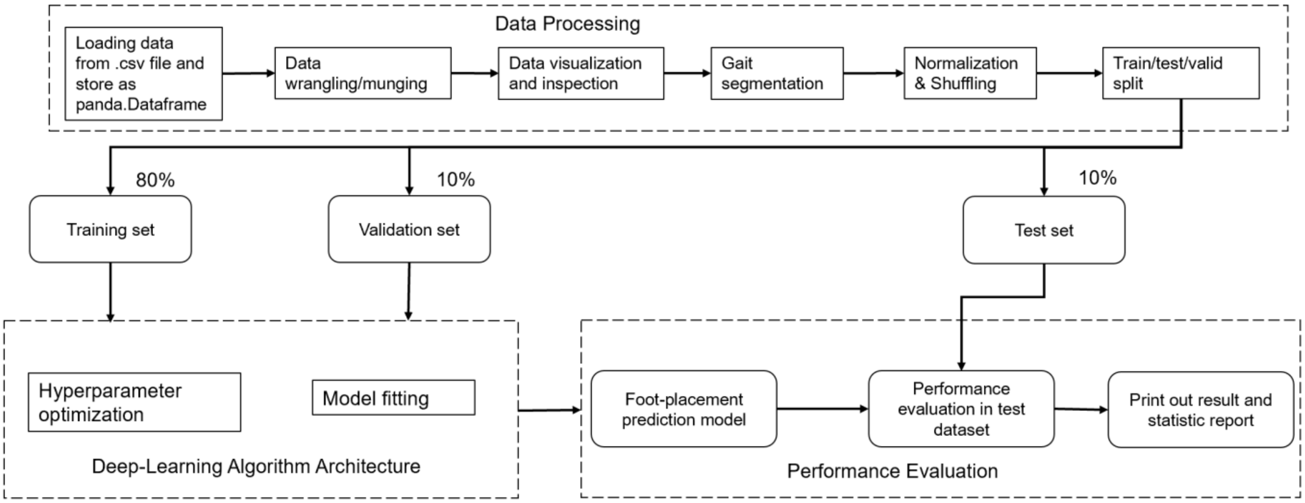

Data Processing
Loading data from .csv file and store as panda.Dataframe
Data wrangling/munging
Data visualization and inspection
Gait segmentation
Normalization & Shuffling
Train/test/valid split
80%
10%
10%
Training set
Validation set
Test set
Hyperparameter optimization
Model fitting
Deep-Learning Algorithm Architecture
Foot-placement prediction model
Performance evaluation in test dataset
Print out result and statistic report
Performance Evaluation

Figure 9: The overview of training pipeline, data processing, model training, and performance evaluation

relevant to what is presently translating and diminish the other. Thus, the model can process long sequence data more effectively [38]. There are many variants of attention mechanisms created in recent years [39], [40], but the one that we implemented for foot placement prediction is adapted based on Bahdanau's version described in [38].

4.6. Proposed Network Architecture

The overall pipeline of our project consists of three components: data processing, fitting deep-learning model, and performance evaluation, as shown in Figure 9.

Data Processing:

The data processing phase will perform a sequence of tasks, including data loading, manipulating data into the desired format, visualization and inspection, gait segmentation, normalization, and train-valid-test splitting with an 80-10-10% ratio. At the end of data processing, the training and validation data will be well-prepared to feed into our deep learning model, and the testing dataset is ready for performance evaluation as well.

Deep-learning Algorithm Architecture:

For this project, we created 4 different neural network architectures for comparison.

Model 1: LSTM

The first model that we implemented contains a single LSTM layer with 128 cells, and the output of the LSTM layer directly flows to the fully connected layer with 2 units. The stacked LSTM network architecture is shown in Figure 10 a), and the model summary produced with TensorFlow is shown in Figure 11.

Model 2: Bi-LSTM

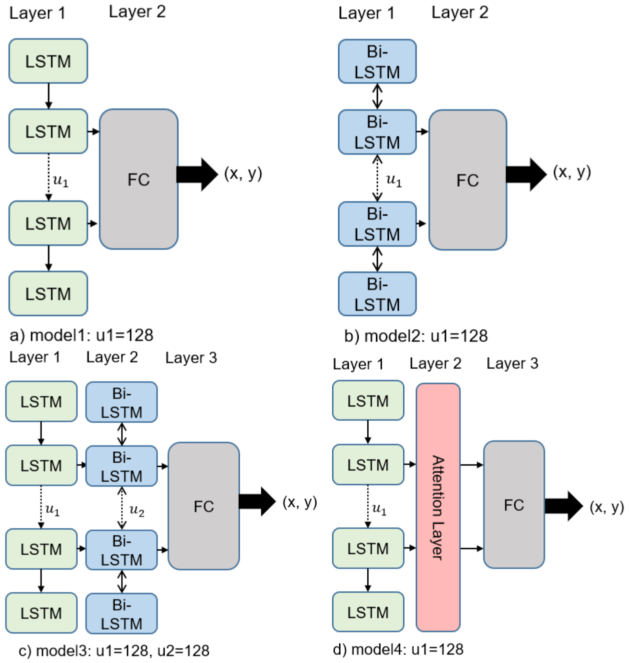
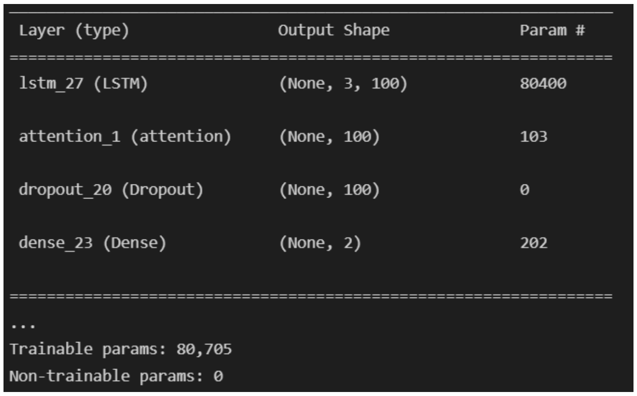


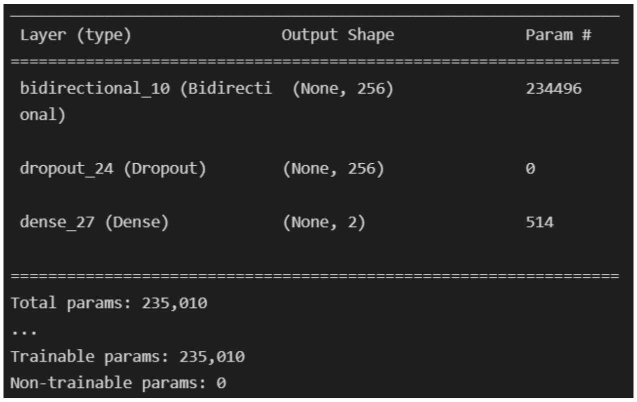
Figure 10: Four proposed neural network architectures, and u_i is the number of cells in layer i. a) Model 1: a single LSTM layer with 128 cells. b) Model 2: single Bi-LSTM layer with 128 cells. c) Model 3: an LSTM and a Bi-LSTM layer with both 128 cells. d) Model 4: an attention layer added to model 1.

Even though our foot placement prediction model might not require the information for future gait trajectory, we still implemented a stacked Bi-LSTM network architecture for verifying its validity and comparing it to other models. The network architecture is shown in Figure 10 b), and the model summary produced with TensorFlow is shown in Figure 12.



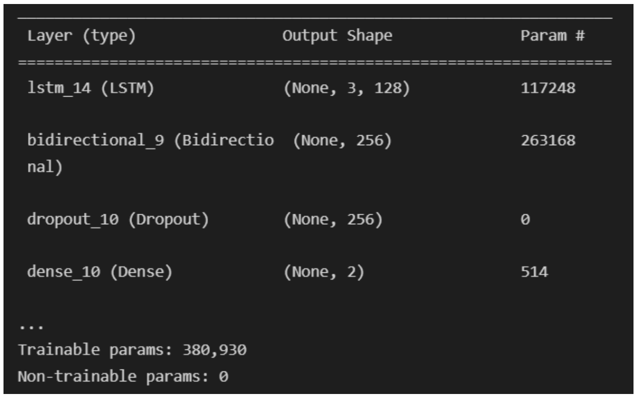
```
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_27 (LSTM)              (None, 3, 100)            80400

 attention_1 (attention)     (None, 100)               103

 dropout_20 (Dropout)        (None, 100)               0

 dense_23 (Dense)            (None, 2)                 202

=================================================================
...
Trainable params: 80,705
Non-trainable params: 0
```

Figure 11: Model 1's model summary



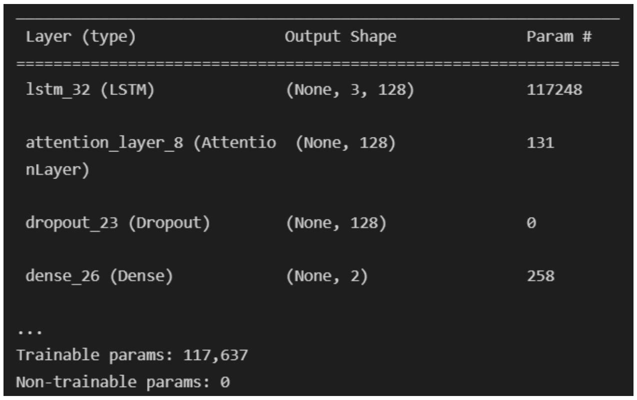
```
 Layer (type)                Output Shape              Param #
=================================================================
 bidirectional_10 (Bidirecti  (None, 256)              234496
 onal)

 dropout_24 (Dropout)        (None, 256)               0

 dense_27 (Dense)            (None, 2)                 514

=================================================================
Total params: 235,010
...
Trainable params: 235,010
Non-trainable params: 0
```

Figure 12: Model 2's model summary



```
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_14 (LSTM)              (None, 3, 128)            117248

 bidirectional_9 (Bidirectio  (None, 256)              263168
 nal)

 dropout_10 (Dropout)        (None, 256)               0

 dense_10 (Dense)            (None, 2)                 514

...
Trainable params: 380,930
Non-trainable params: 0
```

Figure 13: Model 3's model summary



```
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_32 (LSTM)              (None, 3, 128)            117248

 attention_layer_8 (Attentio  (None, 128)              131
 nLayer)

 dropout_23 (Dropout)        (None, 128)               0

 dense_26 (Dense)            (None, 2)                 258

...
Trainable params: 117,637
Non-trainable params: 0
```

Figure 14: Model 4's model summary

Model 3: LSTM + Bi-LSTM

The third network architecture is composed of a layer of LSTM and a layer of Bi-LSTM with 128 and 128 cells respectively. The network architecture for model 3 is shown in Figure 10 c), and its corresponding model summary produced with TensorFlow is shown in Figure 13.

Model 4: LSTM + Attention

In model 4, the attention layer is inserted at the end of the LSTM layer. The attention layer can be thought of as a "single unit Dense layer" and has 131 trainable parameters in total, where a length of 128."attention weight" vector matched to the output size of LSTM, and 3 bias terms matched to the 3 data sequences for each gait trajectory. The network architecture is shown in Figure 10 d), and its corresponding model summary is in Figure 14.

Performance Evaluation:

All four models described above are trained with mean square error (MSE) as a loss function, shown in equation (1). During the training process, an early stop criterion was applied to monitor the validation loss, so that the training process would stop if there were not any improvements for the next 10 epochs. The model with the lowest validation loss throughout the training will be the final one used for testing. The validation sets were specifically used for evaluating the model during hyperparameter tuning, model fitting, and model selection, and the test set is the data that has not been seen by the algorithm and is what will be used for evaluating the generalization performance of the model. While evaluating the performance on the test set, there are two loss metrics were used, mean absolute error (MAE) and mean squared error (MSE), which are shown in Equations (1) and (2):

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(Y_i - \widehat{Y}_i\right)^2 \qquad (1)$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|Y_i - \widehat{Y}_i\right| \qquad (2)$$

where Y_i and $\widehat{Y}_i$ are the ground truth position and predicted position for ith gait trajectory, and n is the total number of gait trajectories of the given dataset.

4.7. Training and Hyperparameter Optimization Details

All present network was trained by using stochastic gradient descent (SGD) optimizer [41] with learning rate and momentum at 1e-6 and 0.9, and the number of epochs and batch size were 200 and 8, respectively. The dropout rate of all dropout layers was set at 0.2. The network was implemented by using Python (https://www.python.org/, accessed on 15 Dec 2022) v3.9.12 and TensorFlow (https://www.tensorflow.org/, accessed on 15 Dec 2022) v2.9.1 on a Windows 10 computer with an Intel Core i7-11800H @ 2.3GHz, an 8 GB memory chip (GDDR6 SDRAM), and a graphics card (GeForce RTX 3080).

5. Result

All four models were evaluated on the same test set, and their results are shown in Table 1.

Model	MAE loss	MSE (RMSE) loss	Time ms/step	Model size
LSTM	10.53	132.03 (11.49)	**29**	117,506
Bi-LSTM	10.89	139.19 (11.80)	41	235,010
LSTM + Bi-LSTM	11.08	143.19 (11.97)	33	**380,930**
LSTM + Attention	**9.62**	**109.62 (10.47)**	**29**	117,637

Table 1: Performance of the deep learning models on the test dataset.

Figure 15 shows the mean square loss on training and validation set throughout the training process, and they were plotted with blue and orange colors respectively.

6. Limitation and Analysis

According to the result in Table 1, the RMSE for LSTM and Bi-LSTM is 11.49 and 11.80 respectively, and Bi-LSTM did not show an advantage over LSTM. Instead, adding the Bi-LSTM layer on top of LSTM raise the RMSE to 11.97, and this deteriorated the performance of prediction on the test set. For the last model, model 4, the Attention Layer was added to the top of the LSTM layer. By comparing the result of model 4 and model 1, adding the attention layer reduced the prediction error without much increase in the network size.

Figure 15 examines the loss on training and validation set during the training process, besides certain fluctuations presented in the training loss of model 1, all models that we trained did not appear overfitting.

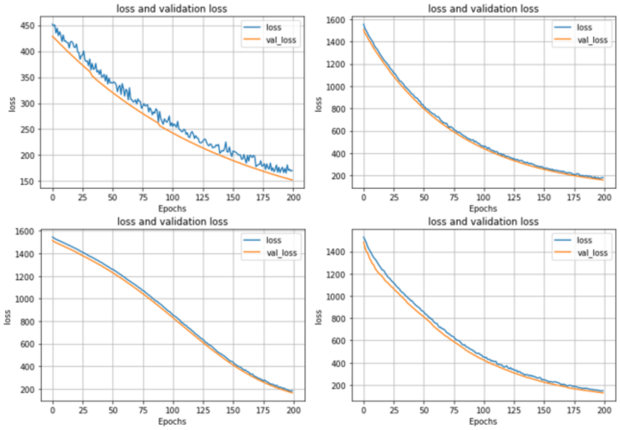


Figure 15: Training loss vs validation loss on four models. a) model 1: top left. b) model 2: top right. c) model 3: bottom left. d) model 4: bottom right.

Besides all observations shown in the result, there are some limitations or potentialities that can be improved in the future. First, the dataset that we used in this project has one specific walking speed, 0.4m/s, and this is not sufficient if we desired a learning model with high generalization ability and being robust to various walking conditions, such as walking uphill/downhill, and walking upstairs/downstairs. Second, all four models implemented in this project are simply for verifying the validity of the attention mechanism and measuring the effectiveness of adding an attention mechanism to the LSTM network architecture on the food placement prediction task. A more serious version of attention-based network architecture can be implemented in the future for achieving better performance, and other attention mechanisms such as Luong's version attention layer [40], Self-attention, and Multi-Headed attention [39]are also worth investigating. Third, hyperparameter tuning. Due to time constraints, we did not spend a huge amount of time fine-tuning the hyperparameter and the selection of various loss functions or optimizers, and we believe there is spacious room for finding the optimal set of hyperparameters. Last, a better gait segmentation approach. The current gait segmentation methods are software based only, and this might not be perfect, especially when multiple activities (e.g., running, stair ascent/descent, etc) are involved. A more accurate and precise gait segmentation might require the support of hardware devices (e.g., foot-switches device) to detect and record the foot position at toe-off and heel-strike events.

7. Conclusion

Our outcome has demonstrated that adding an attention mechanism to the LSTM neural network architecture can make the model to be more precise in food placement prediction. After adding the attention layer to the LSTM network, the MAE, MSE, and RMSE prediction error has reduced by about 8.64%, 16.97%, and 8.88% respectively.

Compared to the result that was produced with the Bayesian inference probability approach in [6], 8.85 to 12.39 RMSE varied on different test conditions, our result is comparable to the previous state-of-the-art performance and has huge potential to improve in the future. In addition, such an approach that we proposed in this study can be generalized to any walking-aid robot, and the experimental result that we collected can be useful for other researchers to design the robot to be more intelligent and comfortable for the patients.

References

[1] W. A. Sparrow and O. Tirosh, “Gait termination: a review of experimental methods and the effects of ageing and gait pathologies,” *Gait Posture*, vol. 22, no. 4, pp. 362–371, 2005, doi: 10.1016/j.gaitpost.2004.11.005.

[2] W. Huo, S. Mohammed, J. C. Moreno, and Y. Amirat, “Lower Limb Wearable Robots for Assistance and Rehabilitation: A State of the Art,” *Ieee Syst J*, vol. 10, no. 3, pp. 1068–1081, 2016, doi: 10.1109/jsyst.2014.2351491.

[3] A. Patla and S. Rietdyk, “Visual control of limb trajectory over obstacles during locomotion: effect of obstacle height and width,” *Gait Posture*, vol. 1, no. 1, pp. 45–60, 1993, doi: 10.1016/0966-6362(93)90042-y.

[4] C. Cao, J. A. Ashton-Miller, A. B. Schultz, and N. B. Alexander, “Abilities To Turn Suddenly While Walking: Effects of Age, Gender, and Available Response Time,” *Journals Gerontology Ser*, vol. 52A, no. 2, pp. M88–M93, 1997, doi: 10.1093/gerona/52a.2.m88.

[5] P. Di *et al.*, “Fall Detection and Prevention Control Using Walking-Aid Cane Robot,” *Ieee Asme Transactions Mechatronics*, vol. 21, no. 2, pp. 625–637, 2016, doi: 10.1109/tmech.2015.2477996.

[6] X. Chen, K. Zhang, H. Liu, Y. Leng, C. Fu, and C. Fu, “A Probability Distribution Model-Based Approach for Foot Placement Prediction in the Early Swing Phase With a Wearable IMU Sensor,” *Ieee T Neur Sys Reh*, vol. 29, pp. 2595–2604, 2021, doi: 10.1109/tnsre.2021.3133656.

[7] F. Sherratt, “Bath Natural Environment HAR Data Set.” Zenodo, Dec. 02, 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4390499

[8] W. Pirker and R. Katzenschlager, “Gait disorders in adults and the elderly,” *Wien Klin Wochenschr*, vol. 129, no. 3–4, pp. 81–95, 2017, doi: 10.1007/s00508-016-1096-4.

[9] L. di Biase *et al.*, “Gait Analysis in Parkinson’s Disease: An Overview of the Most Accurate Markers for Diagnosis and Symptoms Monitoring,” *Sensors*, vol. 20, no. 12, p. 3529, 2020, doi: 10.3390/s20123529.

[10] J. Dicharry, “Kinematics and Kinetics of Gait: From Lab to Clinic,” *Clin Sport Med*, vol. 29, no. 3, pp. 347–364, 2010, doi: 10.1016/j.csm.2010.03.013.

[11] V. Agostini, G. Balestra, and M. Knaflitz, “Segmentation and Classification of Gait Cycles,” *Ieee T Neur Sys Reh*, vol. 22, no. 5, pp. 946–952, 2014, doi: 10.1109/tnsre.2013.2291907.

[12] M. Gadaleta, G. Cisotto, M. Rossi, R. Z. U. Rehman, L. Rochester, and S. D. Din, “Deep Learning Techniques for Improving Digital Gait Segmentation,” *2019 41st Annu Int Conf Ieee Eng Medicine Biology Soc Embc*, vol. 00, pp. 1834–1837, 2019, doi: 10.1109/embc.2019.8856685.

[13] A. Prado, X. Cao, M. T. Robert, A. M. Gordon, and S. K. Agrawal, “Gait Segmentation of Data Collected by Instrumented Shoes Using a Recurrent Neural Network Classifier,” *Phys Med Rehabil Cli*, vol. 30, no. 2, pp. 355–366, 2019, doi: 10.1016/j.pmr.2018.12.007.

[14] K. Zhang, J. Wang, C. W. de Silva, and C. Fu, “Unsupervised Cross-Subject Adaptation for Predicting Human Locomotion Intent,” *Ieee T Neur Sys Reh*, vol. 28, no. 3, pp. 646–657, 2020, doi: 10.1109/tnsre.2020.2966749.

[15] M. Leshno, V. Ya. Lin, A. Pinkus, and S. Schocken, “Multilayer feedforward networks with a nonpolynomial activation function can approximate any function,” *Neural Networks*, vol. 6, no. 6, pp. 861–867, 1993, doi: 10.1016/s0893-6080(05)80131-5.

[16] G. E. Hinton, “Connectionist learning procedures,” *Artif Intell*, vol. 40, no. 1–3, pp. 185–234, 1989, doi: 10.1016/0004-3702(89)90049-0.

[17] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2017.

[18] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, “Meta-Learning in Neural Networks: A Survey,” *Ieee T Pattern Anal*, vol. 44, no. 9, pp. 5149–5169, 2022, doi: 10.1109/tpami.2021.3079209.

[19] Q.-J. Zhang, K. C. Gupta, and V. K. Devabhaktuni, “Artificial Neural Networks for RF and Microwave Design—From Theory to Practice,” *Ieee T Microw Theory*, vol. 51, no. 4, p. 1339, 2003, doi: 10.1109/tmtt.2003.809179.

[20] S. M. Alfayeed and B. S. Saini, “Human Gait Analysis Using Machine Learning: A Review,” *2021 Int Conf Comput Intell Knowl Econ Iccike*, vol. 00, pp. 550–554, 2021, doi: 10.1109/iccike51210.2021.9410678.

[21] B. Pogorelc and M. Gams, “Diagnosing health problems from gait patterns of elderly,” *2010 Annu Int Conf Ieee Eng Medicine Biology*, vol. 2010, pp. 2238–2241, 2010, doi: 10.1109/iembs.2010.5627417.

[22] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to Sequence Learning with Neural Networks,” *Arxiv*, 2014.

[23] A. Graves, A. Mohamed, and G. Hinton, “Speech Recognition with Deep Recurrent Neural Networks,” *Arxiv*, 2013, doi: 10.48550/arxiv.1303.5778.

[24] A. Karpathy and L. Fei-Fei, “Deep Visual-Semantic Alignments for Generating Image Descriptions,” *Arxiv*, 2014, doi: 10.48550/arxiv.1412.2306.

[25] J. Donahue *et al.*, “Long-term Recurrent Convolutional Networks for Visual Recognition and Description,” *Arxiv*, 2014, doi: 10.48550/arxiv.1411.4389.

[26] Y. Bengio, P. Simard, and P. Frasconi, “Learning long-term dependencies with gradient descent is difficult,” *Ieee T Neural Networ*, vol. 5, no. 2, pp. 157–166, 1994, doi: 10.1109/72.279181.

[27] J. Peng, X. Zhang, and H. Li, “Advances in Asian Mechanism and Machine Science, Proceedings of IFToMM Asian MMS 2021,” *Mech Mach Sci*, pp. 680–690, 2021, doi: 10.1007/978-3-030-91892-7_65.

[28] J. Lee, W. Hong, and P. Hur, “Continuous Gait Phase Estimation Using LSTM for Robotic Transfemoral Prosthesis Across Walking Speeds,” *Ieee T Neur Sys Reh*, vol. 29, pp. 1470–1477, 2021, doi: 10.1109/tnsre.2021.3098689.

[29] I. Kang, D. D. Molinaro, S. Duggal, Y. Chen, P. Kunapuli, and A. J. Young, “Real-Time Gait Phase Estimation for Robotic Hip Exoskeleton Control During Multimodal Locomotion,” *Ieee Robotics Automation Lett*, vol. 6, no. 2, pp. 3491–3497, 2020, doi: 10.1109/lra.2021.3062562.

[30] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[31] H. Sak, A. Senior, and F. Beaufays, “Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition,” *Arxiv*, 2014, doi: 10.48550/arxiv.1402.1128.

[32] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to Sequence Learning with Neural Networks,” *Arxiv*, 2014, doi: 10.48550/arxiv.1409.3215.

[33] A. Karpathy, J. Johnson, and L. Fei-Fei, “Visualizing and Understanding Recurrent Networks,” *Arxiv*, 2015, doi: 10.48550/arxiv.1506.02078.

[34] C. Olah, “Understanding LSTM Networks -- colah’s blog,” Aug. 27, 2015. http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (accessed Nov. 26, 2022).

[35] A. Graves and J. Schmidhuber, “Framewise phoneme classification with bidirectional LSTM and other neural network architectures,” *Neural Networks*, vol. 18, no. 5–6, pp. 602–610, 2005, doi: 10.1016/j.neunet.2005.06.042.

[36] G. Ding, A. Plummer, and I. Georgilas, “Deep learning with an attention mechanism for continuous biomechanical motion estimation across varied activities,” *Frontiers Bioeng Biotechnology*, vol. 10, p. 1021505, 2022, doi: 10.3389/fbioe.2022.1021505.

[37] V. Guimarães, I. Sousa, and M. V. Correia, “A Deep Learning Approach for Foot Trajectory Estimation in Gait Analysis Using Inertial Sensors,” *Sensors Basel Switz*, vol. 21, no. 22, p. 7517, 2021, doi: 10.3390/s21227517.

[38] D. Bahdanau, K. Cho, and Y. Bengio, “Neural Machine Translation by Jointly Learning to Align and Translate,” *Arxiv*, 2014, doi: 10.48550/arxiv.1409.0473.

[39] A. Vaswani *et al.*, “Attention Is All You Need,” *Arxiv*, 2017, doi: 10.48550/arxiv.1706.03762.

[40] M.-T. Luong, H. Pham, and C. D. Manning, “Effective Approaches to Attention-based Neural Machine Translation,” *Arxiv*, 2015, doi: 10.48550/arxiv.1508.04025.

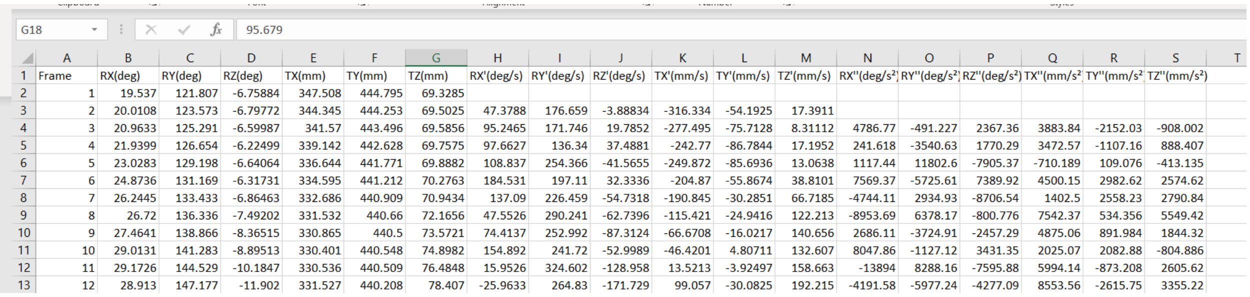
[41] S. Ruder, “An overview of gradient descent optimization algorithms,” *Arxiv*, 2016, doi: 10.48550/arxiv.1609.04747.

[42] J. E. Edelstein, “Geriatric Physical Therapy (Third Edition),” *Part Iv Special Problems Interventions*, no. Arthritis Rheum592008, pp. 412–425, 2012, doi: 10.1016/b978-0-323-02948-3.00031-6.

[43] V. Rai, A. Sharma, P. Preechayasomboon, and E. Rombokas, “Coordinated Movement for Prosthesis Reference Trajectory Generation: Temporal Factors and Attention,” *2020 8th Ieee Ras Embs Int Conf Biomed Robotics Biomechatronics Biorob*, vol. 00, pp. 939–945, 2020, doi: 10.1109/biorob49111.2020.9224435.

Appendix A: Data Collection Plan (Draft)

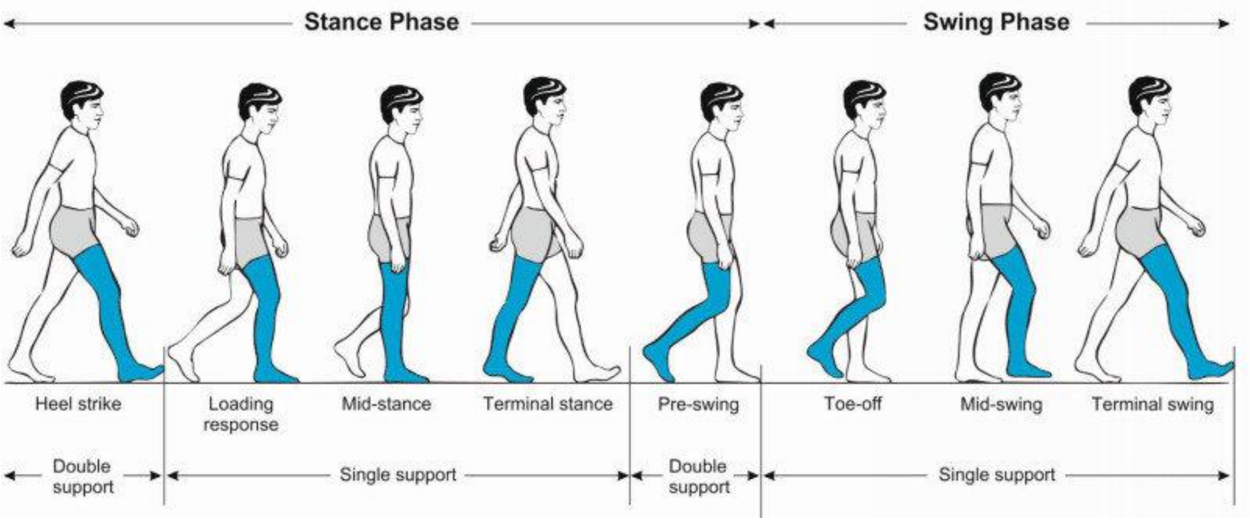
- Format:



G18 95.679

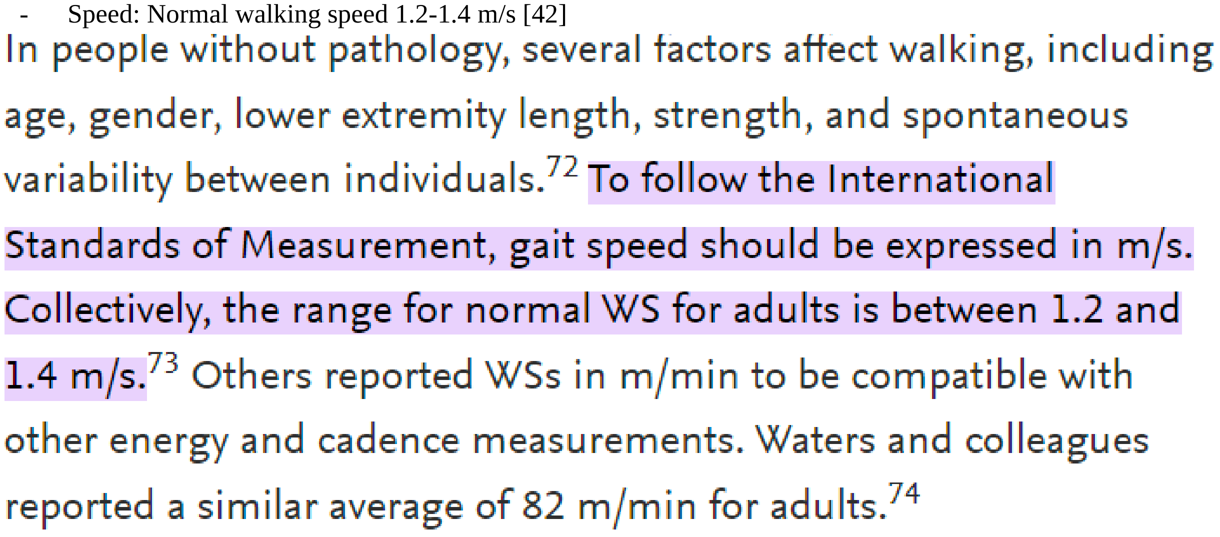
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Frame	RX(deg)	RY(deg)	RZ(deg)	TX(mm)	TY(mm)	TZ(mm)	RX'(deg/s)	RY'(deg/s)	RZ'(deg/s)	TX'(mm/s)	TY'(mm/s)	TZ'(mm/s)	RX''(deg/s^2	RY''(deg/s^2	RZ''(deg/s^2)	TX''(mm/s^2	TY''(mm/s^2	TZ''(mm/s^2)	
2	1	19.537	121.807	-6.75884	347.508	444.795	69.3285													
3	2	20.0108	123.573	-6.79772	344.345	444.253	69.5025	47.3788	176.659	-3.88834	-316.334	-54.1925	17.3911							
4	3	20.9633	125.291	-6.59987	341.57	443.496	69.5856	95.2465	171.746	19.7852	-277.495	-75.7128	8.31112	4786.77	-491.227	2367.36	3883.84	-2152.03	-908.002	
5	4	21.9399	126.654	-6.22499	339.142	442.628	69.7575	97.6627	136.34	37.4881	-242.77	-86.7844	17.1952	241.618	-3540.63	1770.29	3472.57	-1107.16	888.407	
6	5	23.0283	129.198	-6.64064	336.644	441.771	69.8882	108.837	254.366	-41.5655	-249.872	-85.6936	13.0638	1117.44	11802.6	-7905.37	-710.189	109.076	-413.135	
7	6	24.8736	131.169	-6.31731	334.595	441.212	70.2763	184.531	197.11	32.3336	-204.87	-55.8674	38.8101	7569.37	-5725.61	7389.92	4500.15	2982.62	2574.62	
8	7	26.2445	133.433	-6.86463	332.686	440.909	70.9434	137.09	226.459	-54.7318	-190.845	-30.2851	66.7185	-4744.11	2934.93	-8706.54	1402.5	2558.23	2790.84	
9	8	26.72	136.336	-7.49202	331.532	440.66	72.1656	47.5526	290.241	-62.7396	-115.421	-24.9416	122.213	-8953.69	6378.17	-800.776	7542.37	534.356	5549.42	
10	9	27.4641	138.866	-8.36515	330.865	440.5	73.5721	74.4137	252.992	-87.3124	-66.6708	-16.0217	140.656	2686.11	-3724.91	-2457.29	4875.06	891.984	1844.32	
11	10	29.0131	141.283	-8.89513	330.401	440.548	74.8982	154.892	241.72	-52.9989	-46.4201	4.80711	132.607	8047.86	-1127.12	3431.35	2025.07	2082.88	-804.886	
12	11	29.1726	144.529	-10.1847	330.536	440.509	76.4848	15.9526	324.602	-128.958	13.5213	-3.92497	158.663	-13894	8288.16	-7595.88	5994.14	-873.208	2605.62	
13	12	28.913	147.177	-11.902	331.527	440.208	78.407	-25.9633	264.83	-171.729	99.057	-30.0825	192.215	-4191.58	-5977.24	-4277.09	8553.56	-2615.75	3355.22	

- Length: > 5000 Steps, each step consists of a complete stance and swing phase as shown below



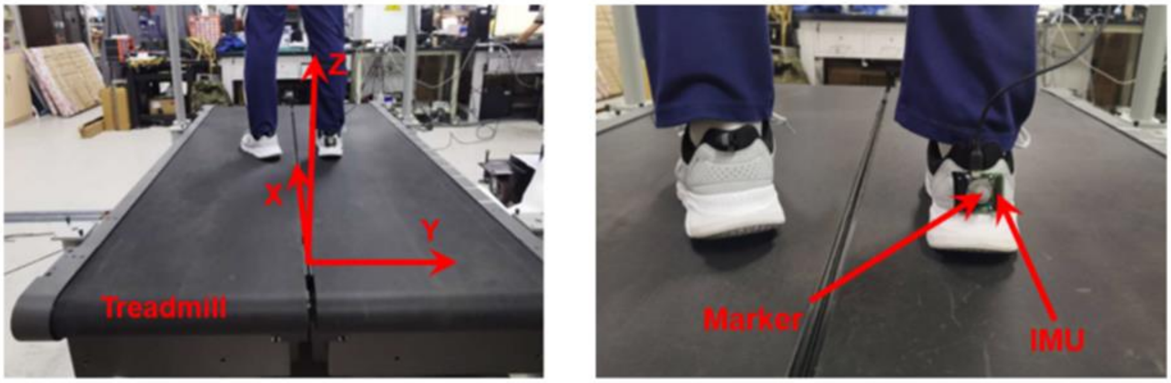
[8]:

- Speed: Normal walking speed 1.2-1.4 m/s [42]



In people without pathology, several factors affect walking, including age, gender, lower extremity length, strength, and spontaneous variability between individuals.[72] To follow the International Standards of Measurement, gait speed should be expressed in m/s. Collectively, the range for normal WS for adults is between 1.2 and 1.4 m/s.[73] Others reported WSs in m/min to be compatible with other energy and cadence measurements. Waters and colleagues reported a similar average of 82 m/min for adults.[74]

- Subject: Normal and Healthy human
- Marker position: Right heel [6]



(a) The treadmill and the coordinate system

(b) The motion capture marker and IMU are attached to the right heel of a subject